AALTO-YLIOPISTO
TEKNILLINEN KORKEAKOULU
Elektroniikan, tietoliikenteen ja automaation tiedekunta

Antti Korhola

HAMMASHOITOKONEEN TESTAUS- JA VIANETSINTÄOHJELMA

Diplomityö, joka on jätetty opinnäytteenä tarkastettavaksi diplomi-insinöörin tutkintoa varten Espoossa 22.4. 2010

Työn valvoja:

Prof. Raimo Sepponen

Työn ohjaaja:

DI Veijo Inkiläinen

AALTO-YLIOPISTO
TEKNILLINEN KORKEAKOULU

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Antti Korhola

Työn nimi: Hammashoitokoneen testaus- ja vianetsintäohjelma

Päivämäärä: 22.4. 2010 Kieli: Suomi Sivumäärä: 10+46

Tiedekunta: Elektroniikan, tietoliikenteen ja automaation tiedekunta

Professuuri: Elektroniikka ja sovellukset Koodi: S3007

Valvoja: Prof. Raimo Sepponen

Ohjaaja: DI Veijo Inkiläinen

Tässä diplomityössä esitellään hammashoitoyksikön testaukseen ja vianpaikannukseen suunniteltu ohjelmisto. Käsiteltävä hammashoitoyksikkö koostuu monesta keskenään kommunikoivasta toimilaitteesta, minkä vuoksi ei ole aina selvää, missä mahdollinen ongelma varsinaisesti syntyy.

Työssä selvitettiin aluksi mitä puutteita eri käyttäjäryhmät näkevät nykytilanteessa ja miten diagnostiikkaan liittyviä asioita on muissa vastaavissa järjestelmissä ratkaistu. Havaintojen pohjalta kokeellisessa osuudessa toteutettiin LabVIEW-kielinen ohjelmisto, joka pyrkii vastaamaan esitettyihin toiveisiin. Ohjelma kerää hoitoyksikön CAN-viestiväylältä tietoja ja esittää ne helpossa muodossa käyttäjälle.

Tuloksena on hoitoyksikön tarpeisiin räätälöity ohjelma, joka helpottaa ongelmien löytämistä ja jota voidaan soveltaa sekä tuotekehitys-, kokoonpano- että huoltoympäristössä.

Avainsanat: hammashoito CAN-väylä toimilaite diagnostiikka vianpaikannus

AALTO-UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

ABSTRACT OF THE
MASTER'S THESIS

Author: Antti Korhola

Title: Testing and Diagnostic Application for Dental Treatment Unit

Date: 22.4. 2010 Language: Finnish Number of pages: 10+46

Faculty: Faculty of Electronics, Communications and Automation

Professorship: Electronics and Metrology Code: S3007

Supervisor: Prof. Raimo Sepponen

Instructor: M.Sc.(Eng.) Veijo Inkiläinen

This thesis presents a software application intended to help in the testing of and diagnosing problems in a dental treatment unit. The unit comprises several subsystems and often it is not clear where exactly some particular problem originates.

Different user groups were asked what kind of diagnostic functionality they would like to have. Based on the findings an application was created in LabVIEW to answer these needs. The purpose of the application is to gather data from the CAN bus of the treatment unit and present it in a readable format.

The resulting tailored communication analysis application makes it possible to more easily pinpoint issues in the research and development, production and maintentance stages of the product.

Keywords: dental treatment CAN-bus diagnostics testing

Esipuhe

Haluan kiittää työn valvojaa, professori Raimo Sepposta ja ohjaajaa, DI Veijo Inkiläistä sekä kaikkia työtovereita opastuksesta ja avusta diplomityön tekemisessä.

Suuri kiitos myös perheelle, ystäville ja muulle lähipiirille tuesta ja kannustuksesta viimeisen puolen vuoden aikana.

Helsinki, 22.4.2010

Antti Korhola

Sisältö

Kuvat

Taulukot

Käsitteet ja lyhenteet

Lyhenteet

BIST engl. Built-In Self-Test, sellainen lähestymistapa diagnostiikkaan jossa testejä rakennetaan mahdollisuuksien mukaan suoraan laitteelle

CAN engl. Controller Area Network, Bosch-yhtiön 1980-luvulla kehittämä kenttäväylä

CANopen yleinen CAN-väylän kanssa käytetty viestiprotokolla

HCI engl. Human-Computer Interaction, ihmisen ja tietokoneen välistä vuorovaikutusta käsittelevä tutkimus

ISO engl. International Organization for Standardization, kansainvälinen standardisointijärjestö

LabVIEW engl. Laboratory Virtual Instrumentation Engineering Workbench, National Instruments-yhtiön graafinen ohjelmointikieli

OBD engl. On-Board Diagnostics, etenkin autojen sisäinen vika- ja muiden tilojen seuranta, jota voidaan sopivalla päätelaitteella lukea standardoidun liittimen kautta

VI engl. Virtual Instrument, LabVIEW-kielinen ohjelma

Sovereign-hoitokoneen mikroprosessoriohjatut piirikortit

ACCU Advanced Centralized Controlling Unit, Sovereign-hoitokoneen linux-pohjainen pääkortti

GUI Graphical User Interface, kosketusnäyttöä ja kaiutinta ohjaava piirikortti

HERCO Headrest Controller, niskatuen moottoreita ja painikkeita ohjaava piirikortti

ICON 3/6 Instrument Controller, 3- tai 6-paikkainen, hoitoinstrumentteja ohjaava piirikortti

MAMCO S/B Multiple Axis Motor Controller, tuolin isoja moottoreita ohjaava piirikortti, joita on hoitokoneessa tavallisesti kaksi kappaletta, toinen istuinosassa (*seat*) ja toinen runko-osassa (*base*)

OLCO Operation Light Controller, hoitovalon ohjainkortti

REFCO A/B Remote Foot Controller, jalkaohjaimen piirikortti, A langaton, B langallinen

RETU Remote Transceiver Unit, langattoman jalkaohjaimen vastaanotinkortti

SLED/SINGLED Single LED Operating Light Controller, ledipohjaisen hoitovalon ohjainkortti

SUCO Suction Controller, imuyksikön toimintaa ohjaava piirikortti

WMC Water Management Controller, vesiyksikön pumppuja ja venttiileitä ohjaava piirikortti

1 Johdanto

Tämän diplomityön tavoitteena on kehittää ratkaisu erään uuden hammashoitoyksikön diagnostiikkatarpeisiin. Hammashoitoyksiköllä tarkoitetaan tässä kokonaisuutta, jossa moottoroidun potilastuolin yhteyteen on integroitu hammashoidossa tarvittavat instrumentit sekä niiden tarvitsemat sähkön-, veden- ja ilmansyöttöjärjestelmät.

Hammashoitoyksikkö koostuu monesta oman mikroprosessorin sisältävästä alijärjestelmästä, minkä vuoksi eri osien välinen viestinvälitys nousee keskeiseen asemaan. Jos järjestelmän toiminnassa esiintyy virheitä, ei välttämättä ole heti ilmeistä, missä kokonaisuuden osassa ongelma varsinaisesti piilee.

Työn aloittamishetkellä hammashoitoyksikön ohjelmistokehitys on edelleen käynnissä ja toteutettavien ominaisuuksien priorisoinnissa etusijalle on asetettu hammaslääkärin työssään eniten tarvitsemat asiat. Tuotannon ja huoltohenkilöstön kaipaamista diagnostiikkaominaisuuksista on toteutettu vain kaikkein välttämättömimmät. Tilanne hidastaa tarpeettomasti laitteiden tuottamista ja tekee huoltotöistä vaivalloisia.

Parhaassa tapauksessa hoitoyksikkö sisältäisi itse kaiken tarpeellisen diagnostiikan, mutta tämä ei ole lyhyellä tähtäimellä välttämättä mahdollista laitteistoresurssien rajallisuuden vuoksi. Tässä diplomityössä tutkitaankin tapoja laatia ulkoinen, esimerkiksi kannettavalla tietokoneella ajettava, apuohjelmisto joka mahdollistaa hoitoyksikön toiminnan tarkemman analyysin vikatilanteissa.

Koska aiemmat hammashoitoyksiköt ovat olleet elektroniikan osalta paljon yksinkertaisempia, kovin paljon valmista materiaalia ei ole käytettävissä apuohjelmiston pohjaksi. Aluksi onkin tarkasteltava muunlaisia useita alijärjestelmiä sisältäviä laitteita ja selvittää miten diagnostiikka on niissä toteutettu. Tällaisia järjestelmiä ovat esimerkiksi autot, tietoliikennejärjestelmät ja monenlaiset teollisuudessa käytettävät laitekokonaisuudet.

Tutkimuksen tavoitteena tässä diplomityössä on siis

- selvittää ja ryhmitellä yleisimmät tuotanto- ja huoltohenkilöstön kohtaamat ongelmat
- selvittää miksi ja miltä osin miltä osin nykyiset työkalut eivät vastaa tarpeisiin
- toteuttaa uusi ratkaisu joka kattaa ainakin akuuteimmat ongelmat

Käytännössä ainakin yleisimpien ongelmien paikantamisen tulisi olla mahdollista ilman erityisen syvällistä asiantuntemusta, jotta tuotteiden ylläpito ei tuota kohtuuttomia vaikeuksia ja kuluja. Tärkeää on myös, että kokonaan uudenlaisen ongelman ilmetessä on mahdollista nopeasti laajentaa käytettävissä olevia välineitä kattamaan uusi tilanne, jotta ongelma saadaan selvitettyä pian ja asiakastyytyväisyys säilyy korkeana.

Ohjelmiston suunnittelussa on huomioitava muilta työntekijöiltä saatu palaute ja pyrittävä sellaiseen käytettävyyden tasoon, että ongelmien selvittämisessä on mahdollista keskittyä itse ongelmaan eikä käytettävän työkalun toimintaan. Ominaisuuksia on syytä jaotella eri näkymiin eri käyttäjäryhmille ja sijoittaa käyttöohjeita myös ohjelman sisälle.

Diplomityö on jaettu seuraaviin osiin

1. Johdanto
2. Lähtökohdat ja tarve, jossa esitellään työn kohteena oleva laite ja sen sisältämää tekniikkaa ja diagnostiikan vaikeuteen johtavia syitä
3. Suunnitelma ja toteutustapa, jossa käydään läpi lähestymistapoja, joita uuden ratkaisun toteuttamisessa voitaisiin noudattaa
4. Tulokset, jossa esitellään työn tuloksena syntyneen ohjelmiston rakenne ja toteutetut ominaisuudet
5. Yhteenveto ja johtopäätökset, jossa arvioidaan työn tuloksia ja jatkokehitysmahdollisuuksia

Diplomityö on tehty Planmeca Oy:lle, joka on maailman kolmanneksi suurin hammaslääketieteen laitevalmistaja. Yhtiön tuotannosta 98 % menee vientiin yli sataan eri maahan.

2 Lähtökohdat ja tarve

Työn tässä osassa perehdyn ensin diagnostiikan käyttöön teollisuudessa yleisellä tasolla selvittääkseni, mitä diagnostiikalta keskimäärin edellytetään, jotta se koetaan hyväksi. Esittelen myös Sovereign-hammashoitoyksikköä ja käsittelen puutteita, joita laitteen tuotekehitys- ja erityisesti tuotantovaiheessa on diagnostiikan osalta havaittu.

2.1 Diagnostiikka teollisuudessa

Sana diagnostiikka on alun perin kreikkaa ja tarkoittaa jonkin asian olemuksen selvittämistä tarkastelemalla [1]. Teknillisissä yhteyksissä diagnostiikalla tarkoitetaan tavallisesti järjestelmässä esiintyvien vikojen havaitsemista ja paikantamista.

Nimenomaan hammashoitoyksiköiden diagnostiikkaan liittyviä lähteitä en onnistunut löytämään, mutta monenlaisia muita järjestelmiä käsitteleviä kyllä. Hyviä artikkeleita löytyi esimerkiksi tietoliikennetekniikan alalta, jossa edellytetään nykyään erittäin luotettavia ja korkealaatuisia verkkoyhteyksiä. Artikkelissa *Linking Diagnostic Software to Hardware Self-Test in Telecom Systems* määritellään hyvin, että toimivan diagnostiikan tulisi mahdollistaa vian paikantaminen sellaisella tarkkuudella, että vika voidaan korjata mahdollisimman vähillä toimenpiteillä. Käytännössä tämä tarkoittaa vian rajaamista pienimpään kentällä päivitettävään osaan, esimerkiksi yksittäiseen piirikorttiin tai moduuliin. Jos vikaa ei kyetä näin rajaamaan, saatetaan korjaustöissä päätyä vaihtamaan myös täysin toimivia osia, mikä nostaa korjauskustannuksia tarpeettomasti. [2]

Artikkelissa *Linking diagnostic software to hardware self-test in telecom systems* esitellään BIST-menetelmä (*built-in self test*), jonka kantavana ajatuksena on rakentaa tarpeelliset testit suoraan laitteeseen piiritasolla. Testejä voidaan näin suorittaa laitteen ollessa jo käytössä eikä mitään ulkoista testijärjestelmää tarvita lainkaan. Vaikka artikkeli liittyykin etupäässä tietoliikennejärjestelmiin, keskeisimmät huomiot ovat hyvin laajennettavissa myös moniin muihin järjestelmiin. Joitain esitetyistä periaatteista voidaan mahdollisesti soveltaa hammashoitoyksikönkin tapauksessa. [3]

Rakenteeltaan lähemmin hammashoitoyksikköä vastaavaa teknologiaa löytyy autoteollisuudesta. Itse asiassa hoitoyksikössä käytettävä kommunikaatioväylä on juuri autoja varten kehitetty CAN-kenttäväylä, jota käsitellään työssä tarkemmin myöhemmin. Autojen elektroniikka jakautuu alijärjestelmiin samaan tapaan kuin hammashoitoyksikössä; yksi järjestelmä ohjaa esimerkiksi moottoria, toinen ilmastointia, kolmas ikkunoita ja niin edelleen. Käyttäjälle esitetään auton toiminnasta vain kaikkein oleellisimmat asiat, kuten nopeus, käytössä oleva vaihde ja joitain muita asioita. Tarkkoja tietoja esimerkiksi moottorin ohjauksesta tai lukkiutumattomien jarrujen antureista tarvitaan vasta autoa korjattaessa tai huollettaessa.

Autoteollisuudessa on pitkään käytetty ns. *on-board diagnostic* -menetelmää, jossa auton alijärjestelmät keräävät tietoa vikatilanteista. Tiedot voidaan jälkikäteen lu-

kea standardoidun liittimen kautta sopivalla päätelaitteella, joka voi olla erillinen kädessä pideltävä malli tai yhteys tietokoneeseen, jossa ajetaan vastaavaa ohjelmaa. Tallennettujen tietojen lukemisen lisäksi on tyypillisesti mahdollista seurata esimerkiksi moottorin polttoaineen syöttöjärjestelmän toimintaa ja muita auton sisäisiä tapahtumia reaaliajassa. OBD-järjestelmä on nykyään Euroopassa ja useimmilla muilla markkinoilla pakollinen, mikä takaa tietyn vähimmäistason kaikkien uusien autojen diagnostiikkatoiminnallisuudelle [4, kohta 14].

Tämän projektin perustavoitteen voisi ajatella olevan eräänlainen OBD-pääte hammashoitoyksikölle. Hoitoyksikön resurssit säästyisivät varsinaiselle hammashoitoon liittyvälle toiminnalle, mutta huoltoa varten laitteeseen voidaan kytkeä helpolla käyttöliittymällä varustettu lisälaite. Jos kyseessä on lisäksi täysin ohjelmistopohjainen ratkaisu, joka ei toimiakseen edellytä mitään lisävarusteita, kustannuksetkin pysyvät alhaisina.

Myös patenttien joukosta löytyi joitain aihetta sivuavia tekstejä. Esimerkiksi yhdysvaltalainen patentti 4,898,263, *Elevator Self-Diagnostic Control System* kuvaa hissinohjausjärjestelmää, joka sisältää diagnostiikan yleisimmille hississä esiintyville virhetilanteille ja osaa kerätä virheet myös lokiin myöhempää tarkastelua varten. Hammashoitoyksikönkin tapauksessa alijärjestelmät käsittelevät itse alimman tason virheet, mutta käyttäjälle ne eivät välttämättä nykyisellään näy, ainakaan riittävän yksityiskohtaisesti. Tämä on yksi asia johon olisi hyvä saada parannusta. [5]

Patentti 5,594,663, *Remote Diagnostic Tool* kuvaa erittäin hyvin sellaista järjestelmää, jota hammashoitoyksikkökin kaipaa. Kuvaustekstissä mainitaan esimerkiksi tarve tarjota asiakkaille edullisia tukipalveluita, jotka eivät edellytä huoltohenkilökunnan käyntiä paikan päällä. Diagnostiikkayhteys voitaisiin muodostaa esimerkiksi puhelinlinjaa pitkin ja ongelmat käsitellä keskitetysti palvelukeskuksessa. Tekstissä huomautetaan myös miten olisi hyödyllistä esittää laitteen toimintaan liittyviä tietoja helposti ymmärrettävällä tavalla ongelmien ratkaisemisen helpottamiseksi. Varsinaiset patentin tekniset toteutustavat eivät juurikaan vastaa hammashoitoyksikön tarpeita, mutta kokonaisuutena patentin kuvausosa tiivistää mainiosti toimivan diagnostiikkaohjelmiston ominaisuudet. [6]

2.2 Sovereign-hammashoitoyksikkö

Hammashoitoyksiköllä tarkoitetaan hammaslääkärin työssään käyttämää järjestelmää, jossa on integroitu yhdeksi laitteeksi hammashoitoon tarvittavat instrumentit paineilma- ja vesisyöttöineen sekä moottoroitu potilastuoli.

2.2.1 Moottoroitu potilastuoli

Potilas istuu tai makaa lääkärikäynnin aikana potilastuolilla. Tuoli on suunniteltu siten, että sen kaikkia liikkeitä ohjataan moottoreilla eikä hammaslääkärin tai hoitajan tarvitse säätää mitään käsin. Liikkeet on lisäksi pyritty automatisoimaan siten, että hoidossa tarvittavat asennot voidaan pääosin tallentaa valmiiksi hoitoyksikön

muistiin ja ajaa automaattisesti yhdellä napinpainalluksella.

Tuolin asennonsäädössä käytetään enimmillään seitsemää moottoria. Osa moottoreista on vaihtovirralla ja osa tasavirralla toimivia. Vaihtovirtamoottoreita käytetään eniten voimaa vaativiin toimintoihin, joita ovat potilastuolin nostaminen pystysuunnassa ja selkänojan asennonsäätö. Tuolin kääntämiseen, selkänojan ja niskatuen pituudensäätöön sekä niskatuen tyynyn kahden kulmanivelen säätöön käytetään tasavirtamoottoreita. Lisäksi koko hoitoyksikön kääntämiseen alustallaan on käytössä vielä yksi tasavirtamoottori. Hoitoyksiköstä on asiakkaille tarjolla erilaisia malleja ja mallista riippuen kaikkia moottoreita ei välttämättä asenneta.

Moottoreita ohjataan kolmella ohjainkortilla, jotka sisältävät pääteasteita ja mikroprosessorin. Kukin ohjainkortti mittaa kolmen moottorin asentoanturi- ja virtatietoja ja huolehtii turvaominaisuuksista. Turvaominaisuuksia ovat mm. moottorien virtarajat, anturien toiminnan seuranta ja turvakytkimet, jotka aktivoituvat törmäystilanteissa. Liikekomennot ohjainkorteille tulevat hoitoyksikön pääkortilta käyttöliittymien nappien painallusten perusteella.

2.2.2 Vesi ja paineilma

Hammashoidossa käytettävät instrumentit tarvitsevat sähkön lisäksi usein paineilmaa, alipainetta tai vettä. Näitä varten hoitoyksikkö sisältää lukuisia venttiilejä, pumppuja ja säiliöitä, joiden tilaa olisi ongelmatilanteissa hyvä päästä seuraamaan tarkemmin. Venttiilien toiminnan varmistaminen on tärkeää jo laitteen valmistusvaiheessa, sillä vuotava tai muuten viallinen venttiili voi aiheuttaa merkittäviä ongelmia, jos isompia määriä vettä pääsee turmelemaan laitteen sähköisiä osia. Venttiilit ja releet sijaitsevat usein sen verran syvällä hoitoyksikön sisällä ettei niiden toimintaa voi helposti suoraan seurata.

2.2.3 Käyttöliittymät

Hoitoyksikön toimintoja ohjataan ensisijaisesti graafiselta kosketusnäytöltä sekä jalkaohjaimelta. Lisäksi niskatukea on mahdollista hienosäätää tyynyn yhteydessä olevilla ristiohjaimilla.

Kattavimmat käyttömahdollisuudet löytyvät kosketusnäytöltä. Näyttö on jaettu useampiin alinäkymiin, joilta voidaan ohjata esimerkiksi tuolin liikkeitä, säätää instrumenttien asetuksia, tallentaa käyttäjäkohtaisia asetuksia ja niin edelleen. Kosketusnäyttö sisältää itsessään tarpeellisen ohjelmiston käyttöliittymän esittämiseen, eikä se siis ole ainoastaan pääkortin passiivinen näyttöpääte.

Jalkaohjaimessa on sivu- ja korkeussuunnassa liikuteltava poljin sekä kaksi kaksisuuntaista liukukytkintä ja yksi nelisuuntainen ristiohjain. Poljinta käytetään pääasiassa poran nopeuden säätöön hoitotilanteessa. Jalkaohjaimen kytkimet ja sen poljin vastaavat yhdessä toiminnaltaan ruudulla kulloinkin näkyviä painikkeita, mikä mahdollistaa yleisimpien toimintojen käytön myös pelkällä jalkaohjaimella. Polkimen asento mitataan kapasitiivisella anturilla ja nappien asennot magneettikenttää

seuraavilla Hall-antureilla.

Niskatuen tyynyn kummallakin sivulla on nelisuuntainen ristiohjain joka toimii myös painikkeena. Ohjainta käyttämällä voidaan säätää niskatuen asentoa kuudessa eri liikesuunnassa vastaavalla tavalla kuin kosketusnäytöltäkin.

Lisäksi työskentelyvalaisimen kirkkaus on säädettävissä valoanturin avulla kädenheilautuksin. Hoitoyksikön käyttöliittymien suunnittelulla ja sijoittelulla on pyritty tekemään hygienian ylläpidosta mahdollisimman helppoa.

2.2.4 Pääkortti

Hoitoyksikön aivoina toimii sulautettu tietokone, jolla ajetaan linux-käyttöjärjestelmää. Järjestelmän prosessit käsittelevät käyttöliittymiltä ja muilta toimilaitteilta tulevaa tietoa ja ohjaavat hoitoyksikön toimintoja tilanteen mukaisesti. Pääkortti tunnetaan järjestelmässä nimellä ACCU (Advanced Centralized Controlling Unit) ja se on kytketty suoraan hoitoyksikön virtalähteeseen, jonka kautta kaikki toimilaitekortitkin saavat virtansa. Virtalähdekortilla ei ole omaa älyä, vaan ACCU ohjaa käynnistyttyään sen jännitelähtöjä.

2.2.5 Toimilaitekortit

Sovereign-hoitoyksikössä on tarkasta mallista riippuen kymmenkunta toimilaitekorttia, joista kukin vastaa tietystä toiminnallisuudesta. Toimilaitteita ohjataan keskitetysti pääkortilta. Toimilaitekorttien sijainnit on merkitty karkeasti kuvaan 1. Kaikkia toimilaitteita ei koskaan löydy samasta hoitoyksiköstä; esimerkiksi hoitajan instrumentteja varten asennetaan vaihtoehtoisesti joko SUCO- tai ICON3-kortti.

Hoitoyksikön kosketusnäyttö pitää sisällään mikroprosessorikortin, joka tunnetaan nimellä GUI (Graphical User Interface). Kortin ohjelmisto tunnistaa kosketusnäytön painallukset, piirtää näytölle asianmukaisen grafiikan ja välittää tiedot ACCU-kortille. GUI voi lisäksi tuottaa äänimerkkejä.

Potilastuolin moottorien ohjaamiseen käytetään kahta erilaista korttityyppiä, joista MAMCO (Multiple Axis Motor Controller) voi ohjata sekä tasa- että vaihtovirtamoottoreita ja HERCO (Headrest Controller) vain tasavirtamoottoreita. MAMCO-kortteja hoitoyksikössä on kaksi, HERCO-kortteja yksi.

Jalkaohjaimia on johdollista ja johdotonta mallia. Langallinen laite tunnetaan nimellä REFCO (Remote Foot Controller) B ja langaton REFCO A. Langallinen jalkaohjain toimii CAN-väylässä samoin kuin mikä tahansa muukin toimilaite, langaton tarvitsee tuekseen lähetin-vastaanotinkortin, joka muuntaa radioviestit CAN-liikenteeksi ja päinvastoin. Tämä kortti on nimeltään RETU (Remote Transceiver Unit).

Hammashoidossa tarvittavia instrumentteja, paineilmaa, imua ja vettä ohjaavat kortit ICON (Instrument Controller), WMC (Water Management Controller) sekä SUCO (Suction Controller).

Kuva 1: Sovereign-hammashoitoyksikkö (kuva: Planmeca, tekstit lisätty)

Työskentelyvaloja on kahdenlaisia. Vanhempi on nimeltään OLCO (Operation Light Controller) ja uudempi SLED (Single LED Operating Light Controller).

2.2.6 Viestinvälitys yksikön sisällä

Toimilaitekortit eivät koskaan keskustele suoraan toistensa kanssa, vaan kaikki viestinvälitys tapahtuu aina toimilaitteen ja pääkortin välillä. Toimilaitteen kannalta tilanne vastaa siis kahdenvälistä keskustelua ja muut väylällä näkyvät viestit jätetään huomiotta.

Toimilaitteiden ja pääkortin väliseen viestintään käytetään tähtimäistä Controller Area Network -väylää. CAN-väylä on Boschin 1980-luvulla kehittämä kenttäväylä, joka suunniteltiin mahdollistamaan mikrokontrollerien keskinäinen viestintä ajoneuvoissa. Ratkaisu on osoittautunut varmatoimiseksi ja sitä käytetään nykyään laajalti monissa muissakin sovelluksissa. [7, s. xiii] Vuodesta 1993 CAN-väylän on määrittänyt standardi ISO 11898 ja sittemmin sen uudemmat versiot [8].

Standardi ei määrää tarkemmin väylän sähkömekaanista rakennetta tai liittimiä, mutta edellyttää niiden täyttävän tietyt kriteerit. Puutteet väylän toteutuksessa voivat johtaa epämääräiseen toimintaan. Väylän tiedonsiirtonopeus voi olla korkeimmillaan 1 Mb/s mutta saavutettava huippunopeus putoaa väylän pituuden kasvaessa. [9]

CAN-väylässä viestikehys voi sisältää joko 11- tai 29-bittisen tunnisteen (*identifier*) ja 0-8 tavua varsinaista tietoa. Lisäksi kehykseen kuuluu CRC-tarkistussumma ja muita kehyksen määrittäviä bittejä, jotka on eritelty taulukoihin 1 ja 2. Jos väylällä on ruuhkaa, viestien välinen tärkeysjärjestys määräytyy siten, että prioriteetin saa aina viesti jonka tunniste on alempi luku. [10]

Taulukko 1: *Base*-tyyppisen CAN-viestikehyksen rakenne

Osa	Bittejä	Merkitys
start-of-frame	1	viestin aloitus
identifier	11	yksilöllinen tunniste
remote transmission request	1	
identifier extension bit	1	aina 0 lyhyessä muodossa
reserverd bit	1	varaus, aina 0
data length code	4	viestin pituus (0-8 tavua)
data field	0-8 tavua	viestin sisältö, tavujen määrä riippuu viestin pituudesta
CRC	15	tarkistussumma
CRC delimiter	1	aina 1
ack slot	1	
ack delimiter	1	aina 1
end-of-frame	7	aina 1

CAN-protokollassa on myös muuntyyppisiä viestikehyksiä, mutta niitä ei varsinaisesti hyödynnetä tarkasteltavan hoitoyksikön tapauksessa, joten jätän ne tässä käsitte-

Taulukko 2: *Extended*-tyyppisen CAN-viestikehyksen rakenne

Osa	Bittejä	Merkitys
start-of-frame	1	viestin aloitus
identifier a	11	tunnisteen ensimmäinen osa
substitute remote request	1	aina 1
identifier extension bit	1	aina 1 pitkässä muodossa
identifier b	18	tunnisteen toinen osa
remote transmission request	1	aina 0
reserved bits	2	varaus, molemmat aina 0
data length code	4	viestin pituus (0-8 tavua)
data field	0-8 tavua	viestin sisältö, tavujen määrä riippuu viestin pituudesta
CRC	15	tarkistussumma
CRC delimiter	1	aina 1
ack slot	1	
ack delimiter	1	aina 1
end-of-frame	7	aina 1

lemättä. Se, miten viestien tunnisteita sovelletaan ja miten varsinainen tietokuorma niissä kuljetetaan, on käyttäjän päätettävissä. Nykyään yksi yleisimpiä protokollia on CANopen [7, s. xv-xvii], mutta sitä ei aikanaan valittu Sovereign-projektiin, vaan viestintään kehitettiin oma ratkaisu.

2.2.7 Kaupalliset CAN-analysaattorit

CAN-väylän yleisyyden ansiosta kaupallisesti on saatavilla paljon laitteita sen analysoimiseen, eikä laitteiden hintakaan ole ongelmallisen korkea. Pelkän CAN-liikenteen seuranta sellaisenaan, ilman viestien tulkitsemista, ei kuitenkaan paljasta kuin aivan alimman tason ongelmia, esimerkiksi väylän ylikuormittumistilanteita, rikkonaisia johtoja tai muuta vastaavaa.

Tarkemmin järjestelmän toimintaan pääsee kiinni purkamalla viestien sisällön luettavaan muotoon. Tällaiseen soveltuvaa laitetta ei ole kaupallisesti mahdollista ostaa, koska Sovereign-hoitoyksikön viestit noudattavat yhtiön sisällä määriteltyä suljettua protokollaa. Tästä huolimatta joitain olemassaolevien ohjelmistojen ominaisuuksia on hyödyllistä huomioida; esimerkiksi viestien suodatusmahdollisuus olisi hyvä saada uuteen ohjelmistoon mukaan, samoin viestien tallennus tiedostoon aikaleimoin.

2.2.8 Hoitoyksikössä käytetty viestiprotokolla

Sovereign-hoitoyksikön tapauksessa pitkää ja lyhyttä kehyksen muotoa käytetään siten, että tunnisteen ensimmäiset 11 bittiä varataan viestinumerolle ja loput 18 toimilaitteen sarjanumerolle. Pidempää viestikehystä käytetään tavallisesti vain käynnistyksen yhteydessä. Sarjanumerolle varattu 18 bitin tila riittää $2^{18} = 262\,144$ laiteyksilölle.

Taulukko 3: Osoiteavaruuden käyttö

Alue		Käyttötarkoitus
0	- 9	turvaominaisuudet
10	- 39	ei käytössä
40	- 60	debug-viestit 1
61	- 127	reset ja reset acknowledge -viestit
128	- 1200	toimilaitteiden viestit
1201	- 1967	ei käytössä
1968	- 2031	debug-viestit 2

Käytössä oleva viestiavaruuden koko on 2032 ($2^{11} = 2048$, mutta ylimmät 16 tunnistetta ovat kiellettyjä) ja se on jaettu taulukon 3 mukaisella tavalla. Turvaominaisuuksiin liittyvät viestit saavat korkeimman prioriteetin), sitten seuraavat toimilaitteiden käynnistysviestit ja lopuksi käytönaikaiseen toimintaan liittyvät viestit.

Eri toimilaitteiden keskinäinen tärkeysjärjestys määräytyy niiden käynnistymisjärjestyksen mukaan. Tärkeysjärjestyksellä ei ole laitteiden toiminnan kannalta varsinaista merkitystä, mutta se hankaloittaa viestien seurantaa, sillä tunnisteet saattavat vaihtua joka käynnistyksen yhteydessä. Jos tämä jäisi huomaamatta, viestejä saatettaisiin tulkita aivan väärällä tavalla.

Osoiteavaruudessa on vielä käyttämättömiä alueita, joita voidaan tarvittaessa hyödyntää muiden alueiden laajentamiseen tai tällä välin debug-tarkoituksiin.

2.2.9 Viestien sisältö

Viestit sisältävät vaihtelevasti erilaisia tietoja toimilaitteesta ja viestin tunnisteesta riippuen. Tietojen muoto on useimmiten joko bittikenttä, joka sisältää kyllä/ei-tyyppisiä tietoja, tai 1-2 tavun mittainen kokonaisluku. Kuhunkin viestiin mahtuu enimmillään kahdeksan tavua tietoa, mutta kaikkia viestejä ei toki ole pakattu täyteen.

Tyypillinen viesti voisi olla esimerkiksi jalkaohjaimen lähettämä tilatieto, jonka sisältö on eritelty taulukossa 4.

Viestejä on hoitoyksikössä käytössä satoja ja usein yksittäinen viesti sisältää tietoja

Taulukko 4: REFCO A:n eräs viesti

Tavu	Muoto	Merkitys
1	bittikenttä	polkimen poikkeutussuunta, turvakytkimen asento, varoitus virheellisestä kalibroinnista ym.
2-3	16-bittinen luku	polkimen vaakasuuntainen asentotieto välillä 0-1000
4-5	16-bittinen luku	polkimen pystysuuntainen asentotieto välillä 0-1000
6	bittikenttä	jalkaohjaimen painikkeiden asennot, kahdeksan suuntaa
7	bittikenttä	laturiin liittyviä tietoja, virransäästötila ym.
8	8-bittinen luku	jalkaohjaimen akuston jännite skaalattuna välille 0-255

monesta eri asiasta. Onkin selvää, ettei raakaa CAN-virtaa seuraamalla voi päätellä kovinkaan paljoa laitteen toiminnasta vaan tiedot on kerättävä ja esitettävä järkevällä tavalla.

2.2.10 Toimilaitteiden osoitteistus

Koska Sovereign-hoitoyksiköstä tarjotaan myyntiin erilaisia konfiguraatioita, on viestien osoitteistus toimilaitteiden kesken päätetty toteuttaa dynaamisesti. Käynnistyksen yhteydessä pääkortti jakaa kullekin toimilaitteelle erillisen 64 viestin laajuisen osoiteavaruuden, jota toimilaite saa käyttää. Tämä osoiteavaruus voi samankin laitteen eri käynnistyskerroilla vaihdella, mistä syystä viestien kuuntelu ulkopuolelta käsin ei ole aivan suoraviivaista.

Osoitteiden jako toimii seuraavasti:

1. toimilaite lähettää käynnistyttyään toistuvasti ns. reset-viestiä laitetyyppinsä mukaisesti kiinteällä viestinumerolla; viesti sisältää toimilaitteen ohjelmiston version ja sarjanumeron

2. pääkortti vastaa reset acknowledge-viestillä, joka sisältää osoitteen pääkortin valitsemaan vapaaseen osoitelohkoon; tämänkin viestin numero on kiinteä

3. toimilaite siirtyy normaalin toiminnan tilaan ja lähettää tästedes kaikki viestinsä sille osoitetulla 64 osoitteen lohkolla; sarjanumeroa ei enää lähetetä viestien mukana

Joidenkin laitteiden tapauksessa käytettävissä ollut 64 viestin avaruus ei ole riittänyt, jolloin sitä on laajennettu ottamalla yksi sisältötavuista käyttöön tunnisteen jatkeeksi. Yhdenkin viestin kohdalla tehtynä tämä lisää 256 uutta tunnistetta, mikä jo riittääkin lähes kaikkeen. Näillä niin sanotuilla dynaamisilla viesteillä voi tietysti olla varsinaista sisältöä vain seitsemän tavua kahdeksan sijaan.

Viestiliikenteen seurannassa dynaamiset viestit aiheuttavat jonkin verran lisää työtä, sillä yksin viestin tunnisteen perusteella ei voidakaan enää tietää mihin sisältö liittyy, vaan ensin on tarkistettava myös tunnisteen dynaaminen osuus.

Joissain tapauksissa saman viestin sisältö voidaan myös tulkita eri tavoin riippuen jostakin aikaisemmasta viestistä, mikä asettaa vaatimuksia näiden viestien seuraamiselle. Jos tiedon muotoa kertovaa viestiä ei ole lainkaan vastaanotettu tai hoitokone on mahdollisesti käynnistetty sittemmin uudelleen, täytyy olla erityisen tarkkana tällaisten viestien purkamisen kanssa.

Jotta viestiliikenteestä saisi selvää väylää kuuntelemalla on tiedettävä, mikä osoiteavaruus kuuluu millekin toimilaitteelle. Tarvittavat tiedot voi kerätä kuuntelemalla hoitoyksikön viestiliikennettä alusta lähtien ja taltioimalla muistiin reset acknowledge -viestien sisällön, tai tarkastella pääkortille tallentuvia lokitiedostoja, joista osoitteet käyvät myös ilmi. Näistä ensimmäinen vaihtoehto on käyttökelpoisempi, koska seuranta voidaan automatisoida, mutta ei tietysti onnistu jos hoitoyksikkö on jo käynnissä ennen kuuntelun aloittamista.

2.3 Vikojen paikannus hoitoyksikössä

Hoitoyksikkö esittää nykyisellään yleisimmistä virheistä ilmoituksen käyttäjälle ja pitää toiminnastaan lokia. Varsinaista huoltotilaa ei kuitenkaan vielä ole olemassa.

Pääkortin ohjelmisto tallettaa erityisesti omasta toiminnastaan tarkkoja lokitiedostoja, joista on usein apua ongelmien paikallistamisessa. Lokeja ei kuitenkaan pääse näkemään suoraan hoitoyksikön omalta näytöltä ja niiden tulkinta voi olla työlästä. Lokit myös paisuvat helposti hyvin suuriksi, joten jos virhetilannetta ei voida tutkia välittömästi, siihen liittyvien tietojen perkaaminen tekstimassasta käy pian mahdottomaksi.

Kun hoitoyksikön toiminnassa tapahtuu jokin virhe, esitetään käyttäjälle kosketusnäytöllä virheilmoitus sekä lyhyt kuvaus mahdollisista syistä. Virheen tyypistä riippuen ilmoituksen voi kuitata pois joko heti tai vasta sitten, kun virheen aiheuttaja on poistettu. Ilman tarkempaa tietoa on kuitenkin usein vaikea sanoa, mikä johti virheen syntymiseen.

Mahdollisuudet hyödyntää nykyistä kosketusnäyttöä diagnostiikan käyttöliittymänä ovat melko vähäisiä. Tämä johtuu ennen kaikkea näytön ohjainkortin prosessointitehon ja muistikapasiteetin asettamista rajoista. Näytön resoluutio ei myöskään salli esimerkiksi kovin tarkkojen kuvaajien piirtämistä.

2.4 Kohdattuja ongelmia

Hoitoyksikön kehitystyössä tärkeysjärjestyksessä ensimmäisellä sijalla ovat olleet ominaisuudet, joita hammaslääkäri päivittäisessä työssään tarvitsee. Huoltotoiminnallisuus on jäänyt sikäli taka-alalle, että graafiselta käyttöliittymältä ei ole juurikaan tehtävissä mitään huoltotoimenpiteitä tai nähtävissä tietoja, jotka eivät suo-

raan liity hoitotyöhön. Tämä on tehnyt huoltohenkilöstön työstä vaikeaa tapauksissa, joissa näytölle ilmestyvä virhekoodi ei riitä sinänsä mahdollisesti yksinkertaisenkin ongelman selvittämiseen.

Usein vain yhtiön tuotekehitysosastolla on ollut tarpeelliset resurssit selvittää monimutkaisempia ongelmia, mikä tekee kentällä tapahtuvista huolloista hankalia ja kalliita. Diagnostiikkamahdollisuuksia pitäisikin tuoda paremmin kaikkien hoitoyksiköiden parissa työskentelevien käytettäväksi. Tämä asettaa työkalujen helppokäyttöisyydelle tiukempia vaatimuksia, sillä kaikkien ei voida edellyttää tuntevan hoitoyksikön rakennetta syvällisesti aivan alimmalta tasolta lähtien.

Toisinaan paremmalle diagnostiikalle on ilmennyt tarvetta jo tuotantovaiheessa. Jos hoitoyksiköstä löydetään jokin ongelma vasta lopputestausvaiheessa, saattaa se johtaa työlääseen remonttiin viallisen osakokoonpanon korjaamiseksi. Tehokkaalla diagnostiikkatyökalulla voitaisiin havaita aiemmassa vaiheessa myös sellaisia ongelmia, joiden toimintaa on muutoin mahdollista testata vain lopullisessa koonnoksessa.

2.4.1 Esimerkkejä

Esittelen tässä esimerkkejä tapauksista, joissa nykyisin käytettävissä oleva diagnostiikka ei ole riittänyt ongelman ratkaisemiseen paikan päällä, vaan tutkimus on täytynyt siirtää tuotekehityksen piiriin.

Havaittiin, että potilastuolin moottoroidun niskatuen liike pysähtelee joissain yksilöissä selittämättömästä syystä. Hoitoyksikön virhelokissa on nähtävissä virheilmoitus, jonka mukaan nivelen virtaraja on ylittynyt tai liikeanturissa on vikaa. Niskatukea ei kuitenkaan testissä kuormitettu lainkaan eikä liikeanturina toimivassa potentiometrissä havaittu vikaa.

Pitkällisen tutkimuksen jälkeen huomattiin, että potentiometrille vievän johdon liitin on hyvin heppoinen ja johtimien kontakti irtoaa nivelen ollessa tietyssä asennossa. Ongelma kuitenkin poistuu kun moottoritilan kannen avaa. Ongelma olisi paljastunut helposti, jos potentiometrin jännitteestä olisi voinut piirtää kuvaajan ajan suhteen.

Toinen esimerkki liittyy langattoman jalkaohjaimen akustoon. Osa käyttäjistä ilmoitti, ettei jalkaohjaimen käyttöaika ole tarpeeksi pitkä. Tuotekehityksen omien testien perusteella käyttöaika vaikutti kuitenkin aivan kelvolliselta. Asian selvittämiseen olisi tarvittu tarkempaa tietoa siitä, kuinka aktiivisesti hammaslääkärit esimerkiksi työpäivän tai työviikon aikana jalkaohjainta käyttävät, mutta tällaista tietoa ei ollut olemassaolevista lokitiedostoista mitenkään helposti kerättävissä.

Näihin tapauksiin palataan työssä myöhemmin, kun tarkastellaan lopputuloksia ja arvioidaan, olisiko kunnollinen diagnostiikka auttanut paikantamaan tämänkaltaisia vikoja nopeammin.

2.4.2 Kerättyjä parannusehdotuksia

Muodostaakseni perustellun käsityksen tärkeimmistä ominaisuuksista, joita uuteen ohjelmistoon tulisi saada, tiedustelin Sovereign-hoitoyksiköiden parissa työskenteleviltä heidän mielipiteitään asiasta.

Hoitoinstrumenttien tilaa toivottiin nähtäväksi aikajanalle. Kun hammaslääkäri tai hoitaja ottaa instrumentin käteensä, hoitoyksikkö aktivoi tarpeelliset sähkö-, vesi- ja ilmasyötöt ja näyttää kosketusnäytöllä instrumenttiin liittyvät tiedot ja säätimet. Näiden tietojen tallentaminen auttaisi paikantamaan instrumenttien toiminnassa mahdollisesti esiintyviä vikoja ja antaisi samalla kokonaiskuvan siitä, mitä instrumentteja missäkin hoitotilanteessa käytetään ja millä tavalla. Käytännössä tarvittavien tietojen kerääminen CAN-väylältä voi olla liian vaikeaa, sillä iso osa instrumenttien ohjauksesta tapahtuu syvällä toimilaitteiden ohjelmistojen sisällä eikä näiden viestien tulkitseminen ilman kontekstia ole suoraviivaista.

Hoitoyksikön venttiilien, releiden ja paineantureiden tilojen seuraamismahdollisuutta toivottiin myös yleisesti. Tilojen näkemisestä olisi paljon apua kalibrointi- ja lopputestausvaiheissa. Nämä tiedot on verrattain helppo lukea väylältä, eikä ominaisuuden toteuttamisessa pitäisi olla vaikeuksia. Sama koskee hoitoyksikön vesisäiliön tilaa. Ainoa hankaluus on, että tilatietoja lähetetään väylällä vain niiden muuttuessa, mikä voi tarkoittaa etteivät tiedot ole ajan tasalla jos diagnostiikkaohjelmisto kytketään hoitoyksikköön sen jo ollessa käynnissä.

Potilastuolin moottoreiden asentoja seurataan potentiometreillä. Jos potentiometri on viallinen tai kokonaan rikki, ohjauskorttien ohjelmisto saattaa käyttäytyä arvaamattomasti eikä syntyvistä virheilmoituksista välttämättä selviä, onko vika potentiomterissä, moottorissa vai mekaniikassa. Potentiometrien arvojen näkeminen aikajanalla paljastaisi nopeasti mahdolliset ongelmat, eikä ominaisuutta ole vaikea toteuttaa. Yleisessä tapauksessa asentotietoa tosin lähetetään vain silloin, kun moottoria ajetaan, joten jos arvoja halutaan seurata jatkuvasti, tarvitaan mahdollisesti muutoksia toimilaitteisiin.

Moottorien asentoanturien kalibrointikohtia toivottiin myös jollain tapaa havainnollistettaviksi. Näin olisi mahdollista nähdä ovatko potentiometrit oikeassa asennossa suhteessa mekaniikkaan joutumatta purkamaan hoitoyksikön katteita auki. Tämä on osittain mahdollista toteuttaa, mutta kaikkia kalibrointipisteitä ei ole mahdollista tarkistaa avaamatta ainakin joitain paneeleita.

Moottoreista haluttiin tietää myös niiden ottamat virrat. Virran perusteella voidaan arvioida moottorin kuntoa ja mekaniikan aiheuttamaa liikevastusta. Virta ei ole tieto, jota moottorit tavallisesti lähettävät väylälle, mutta pyydettäessä tieto kyllä saadaan. Ominaisuus on siis toteutettavissa.

CAN-väylällä kulkevia viestejä toivottiin myös tallennettaviksi. Jos kaikki viestit väylältä luetaan, ei niiden tallentaminen tiedostoon ole toki mikään ongelma. Jonkinlaisia suodatustoimintoja täytynee harkita samaan yhteyteen, jotta voidaan erotella tallennettaviksi esimerkiksi vain tiettyä toimilaitetta koskevat viestit.

Langattoman jalkaohjaimen osalta olisi hyödyllistä päästä näkemään ainakin sen akuston jännite, jota ei muutoin ole mahdollista seurata. Lisäksi sekä langallisen että langattoman jalkaohjaimen tapauksessa voisi olla tarpeellista seurata kalibroinnin vaiheita, joista ei myöskään tavallisesti saa mitään palautetta. Molemmat tiedot ovat helposti poimittavissa väylältä.

Koska mahdollisuus muodostaa raaka telnet- tai sarjayhteys pääkortille on luultavasti aina tarpeellinen, näitä ominaisuuksia toivottiin myös yhdistettäväksi muuhun diagnostiikkaan. Pääteohjelmistojen rakentaminen tyhjästä olisi resurssien haaskausta, mutta mahdollisuus laukaista ulkoinen pääteohjelma oikeilla asetuksilla pitäisi olla toteutettavissa vähäisin panostuksin.

3 Suunnitelma ja toteutustapa

Tässä osassa esitellään teknologioita ja lähestymistapoja, joilla diagnostiikkaan soveltuva ohjelmisto voitaisiin mahdollisesti toteuttaa.

3.1 Työn reunaehdot

Jotta tästä projektista saataisiin irti toivottu hyöty, täytyy toteutuksessa pitäytyä tiettyjen reunaehtojen sisällä resurssien ja ajan käytön osalta. Resursoinnin osalta tavoite on, että diagnostiikkaohjelmisto voidaan toteuttaa pääosin erillisenä projektina siten, ettei se edellytä hoitoyksikön muuhun tuotekehitykseen puuttumista.

3.1.1 Päällekkäisen työn välttäminen

Tiedossa on, että hammashoitoyksikköön ollaan tulevaisuudessa toteuttamassa huoltotila, jonka kautta voidaan hoitaa myös suurin osa yleisestä diagnostiikasta. Hoitoyksikön ohjelmointi tapahtuu pääosin C-kielellä, joten LabVIEW-ohjelmista ja niihin käytetystä työpanoksesta ei ole tällöin mitään apua, vaan huoltotilat on toteutettava puhtaalta pöydältä. Tästä johtuen tämän diagnostiikkaprojektin toteuttamisessa ei liene perusteltua käyttää suhteettomasti aikaa sellaisiin asioihin, jotka ovat hoitoyksikköön integroituna hyvin helppoja tehdä, mutta erillisenä ohjelmana vaikeita.

3.1.2 Nopea hyödynnettävyys

Projekti olisi hyvä saada käytettävään kuntoon mahdollisimman pian, jotta ainakin akuuteimpiin tuotannon ja huollon epäkohtiin saadaan parannusta. Ripeys on sikälikin perusteltua että jos projektissa kestää kovin pitkään, voitaisiin resurssit yhtä hyvin tai paremmin siirtää suoraan hoitoyksikön oman diagnostiikkatoiminnallisuuden kehittämiseen.

3.2 Olemassaolevat työkalut

Hoitoyksikön tuotekehitysprosessissa on jo aikaisemmin syntynyt joitain työkaluja, joista voidaan ottaa mallia ja poimia toiminnallisuuksia uuteen ohjelmistoon.

3.2.1 Visual Basic -sovellus

Tarve yksittäisen toimilaitekortin kanssa kommunikointiin tuli ilmi jo hyvin aikaisessa vaiheessa tuotekehitysprosessia, sillä pääkortin ja toimilaitteiden ohjelmistoja kehitettiin rinnakkain. Instrumentteja ohjaavalle ICON-kortille tehtiinkin Microsoftin

Visual Basicilla työkalu, jolla kortin kehitystyötä voitiin viedä eteenpäin pääohjelmistosta riippumatta. Sittemmin samaa työkalua on laajennettu kattamaan muutkin toimilaitekortit.

Sovellus käyttää tietokoneen USB-liitäntään kytkettävää CAN-sovitinta, jonka kautta se muodostaa yhteyden toimilaitekorttiin tai hoitoyksikköön. Ohjelmalla on mahdollista esimerkiksi päivittää toimilaitteiden ohjelmistoja, asettaa korteille sarjanumeroita ja suorittaa kalibrointeja. Käyttöliittymä on jaettu muutamaan kymmeneen ikkunaan joista kukin kattaa jonkin kortin tietyn osatoiminnallisuuden.

Sovellus on tuotekehityksen tarpeisiin hyödyllinen ja toimiva ratkaisu, mutta sisältää valtavan määrän ominaisuuksia joista ei huoltohenkilöstölle ole mitään apua. Lisäksi sovelluksen rakenne on rönsyillyt kehitystyön mukana milloin minnekin, mikä tekee sen käytön opettelemisesta vaikeaa ja edellyttää hoitoyksikön rakenteen hyvää tuntemista. Taustansa vuoksi ohjelmisto soveltuu myös paremmin yksittäisen kortin kanssa toimimiseen, joskin yhteensopivuutta kokonaisen hoitoyksikön seuraamiseen on myöhemmin parannettu. Ohjelmistoa ei myöskään voisi sellaisenaan päästää yleiseen käyttöön, sillä sen avulla on mahdollista vaihtaa laitteiden sarjanumeroita.

Visual Basic on yleiskäyttöön tarkoitettu ohjelmointikieli, joten kaikki oleellinen toiminnallisuus on sen avulla mahdollista toteuttaa. Ohjelmointikielenä se on myös verrattain helppo ja eikä muodosta korkeaa kynnystä uusien ominaisuuksien kehittämiseen vaikka ohjelmistokehittäjä vaihtuisi. Kieli ei kuitenkaan ole mitenkään erityisesti suunnattu teollisuuden käyttötarkoituksiin, joten lähes kaikki toiminnallisuus on tehtävä alusta loppuun itse. [11]

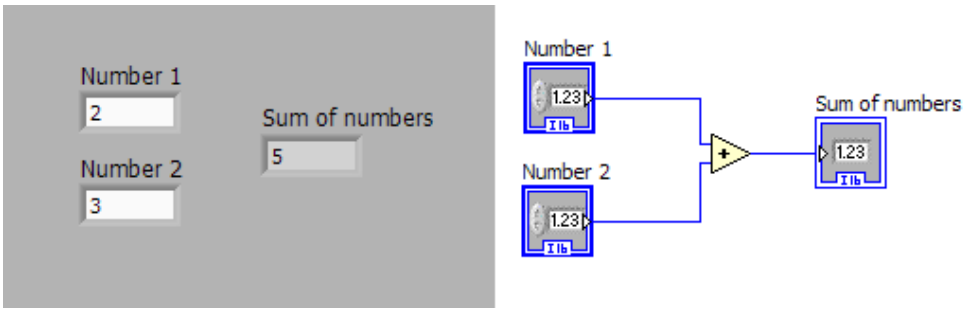
3.2.2 LabVIEW

LabVIEW (*Laboratory Virtual Instrumentation Engineering Workbench*) on graafinen, vuokaavioihin perustuva ohjelmointikieli, joka on suunniteltu erityisesti tutkijoiden ja teollisuuden käyttöön. Sen kantavana ajatuksena on mahdollistaa testijärjestelmien ohjelmointi ilman perinteisen tekstipohjaisen ohjelmoinnin opiskelua. Se sisältää myös valmiiksi paljon funktioita, joita teollisten järjestelmien ohjelmoinnissa tavalisesti tarvitaan. LabVIEW'n ensimmäinen versio julkaistiin 1986. [12]

LabVIEW-kieli tunnetaan myös nimellä G. Ohjelmointi tapahtuu asettelemalla graafisesti kaaviopohjalle valmiita toimintoja. Toimintojen suoritusjärjestys määräytyy yhdistämällä toiminnot langoilla, joita pitkin tieto etenee. Kukin toiminto suoritetaan, kun sen tarvitsemat tiedot ovat kaikki saatavilla. Näin voi tapahtua samanaikaisesti useamman toiminnon kohdalla, jolloin LabVIEW'n sisäänrakennettu aikataulutusjärjestelmä jakaa toiminnoille suoritusaikaa. LabVIEW onkin luontaisesti rinnakkaisprosessointia tukeva kieli. [12]

Käyttöliittymäsuunnittelu on kiinteästi sidottu muun ohjelman kehitykseen. Kullakin ohjelmalla tai aliohjelmalla on kolme pääasiallista komponenttia, jotka ovat vuokaavio, etupaneeli ja liitinpaneeli. Analogia sähkömekaanisiin mittalaitteisiin on siis selvästi esillä ja LabVIEW-ohjelmista käytetäänkin nimitystä *virtual instrument* (VI). Jos ohjelmaa halutaan käyttää toisen ohjelman osana, se voidaan sijoittaa vuo-

kaavioon ja yhdistää muuhun ohjelmaan liitinpaneelinsa välityksellä. Ohjelma voidaan myös esittää etupaneelinsa avulla suoraan käyttäjälle, joka voi näin syöttää ja lukea tietoja. Myös aliohjelmien etupaneeleita voidaan hyödyntää kehitystyössä, vaikka niitä ei lopullisessa ohjelmassa olisikaan tarkoitus käyttäjälle näyttää. [12]



Kuva 2: Yksinkertainen LabVIEW-ohjelma

Kuvassa 2 on hyvin yksinkertainen esimerkki LabVIEW-ohjelmasta. Kuvan vasen puoli esittää ohjelman käyttöliittymää, jossa automaattisesti näkyvät kaikki ohjelman muuttujat. Oikealla puolella on vuokaavio, joka määrää ohjelman toiminnan. Esimerkin tapauksessa lasketaan kaksi muuttujaa yhteen ja esitetään summa kolmannessa.

Kielen graafisen luonteen johdosta kynnys tehdä ohjelmia tai muutoksia ohjelmiin on alhaisempi kuin perinteisillä ohjelmointikielillä. Tämä mahdollistaa toisaalta kielen helpomman käytön ilman muuta ohjelmointikokemusta, mutta voi olla myös petollista, sillä vain ymmärtämällä jonkin verran LabVIEW'n sisäistä toimintaa voi kirjoittaa monimutkaisempia mutta silti tehokkaita ohjelmia. [12]

LabVIEW on ollut Planmeca-yhtiössä käytössä jo entuudestaan tuotannon testijärjestelmissä, joten siitä on valmiiksi jonkin verran kokemuksia. Sitä ei kuitenkaan ole yhtiössä aiemmin sovellettu CAN-väylän lukemiseen joten mitään valmiita osia ei ole suoraan diagnostiikkaa varten käytettävissä.

LabVIEW'n ohella National Instruments tarjoaa myös Test Stand -ohjelmistoa, joka on testien hallinnointiin ja automatisointiin suunniteltu kokonaisuus. Koska LabVIEW ja Test Stand on suunniteltu toimimaan yhdessä, voidaan työmääriä joidenkin ominaisuuksien osalta mahdollisesti yhdistää käyttämällä osia diagnostiikkaohjelmistosta testijärjestelmissä. [13]

Periaatteessa mikään ei estä tekemästä diagnostiikkaohjelmistoa millä tahansa yleisohjelmointikielellä, kuten C:llä, C++:lla tai Javalla. Näillä ei kuitenkaan saavuteta mitään erityistä etua esim. Visual Basiciin nähden, koska suorituskyky ei tällaisessa käyttötarkoituksessa ole kovin merkittävä ongelma. Tämän ja edellämainittujen seikkojen perusteella diagnostiikkaohjelmiston toteutustavaksi valittiin LabVIEW.

3.3 Liityntätapa

Hoitoyksikön viestiliikennettä voi seurata joko suoraan CAN-väylää kuuntelemalla tai pääkortin välityksellä. Molemmilla lähestymistavoilla on hyvät ja huonot puolensa.

3.3.1 Viestien luku suoraan väylältä

CAN-viestejä on mahdollista lukea sopivalla sovittimella suoraan CAN-väylästä. Sovitin muuntaa viestit esimerkiksi USB-liitännässä kuljetettaviksi jolloin niitä voidaan lukea millä tahansa tietokoneella. Näin luettuina viestien ajoitus nähdään erittäin tarkasti, mistä voi olla apua aikakriittisten tapahtumien seurannassa.

Hoitoyksikössä ei ole varsinaista ulkoista liityntää CAN-väylään, koska sellaiselle ei tavallisessa käytössä ole mitään tarvetta. Tämä tarkoittaa että huoltotilanteessa olisi purettava jonkin verran katteita ja suojapeltejä jotta diagnostiikkasovelluksen kytkeminen käy edes mahdolliseksi. Usein huoltotilanteessa kyseisiä peltejä on tosin kaikesta huolimatta avattava, mutta tämä sotii silti jossain määrin diagnostiikan helppokäyttöisyyttä vastaan.

3.3.2 Pääkortin ethernet-liitännän hyödyntäminen

Koska pääkortti vastaanottaa kaikki CAN-viestit, se voi myös melko helposti ohjata ne tekstimuodossa ethernet-väylää pitkin ulkomaailmaan. Tässä menetetään viestien tarkka ajoitus, mutta niitä voidaan vastaanottaa miltei millä hyvänsä tietokoneella ilman uusia lisälaitteita. Tällä hetkellä ethernetin kautta ei myöskään ole mahdollista lähettää viestejä hoitoyksikölle päin, mikä rajoittaa merkittävästi esimerkiksi mahdollisuuksia käynnistää mittauksia diagnostiikkatyökalusta käsin.

3.3.3 Erillinen diagnostiikkarajapinta

Kokonaan oma, diagnostiikkaa varten suunniteltu rajapinta hoitoyksikön ja tietokoneen välille olisi kaikkein kattavin ratkaisu, sillä tällöin myös pääkortin ohjelmistossa tapahtuvat asiat tulisivat seurattaviksi. Lähestymistapa myös erottaisi diagnostiikkaohjelmiston toiminnan CAN-viestien tunnisteista, jotka voivat periaatteessa muuttua minkä tahansa päivityksen yhteydessä. Oman, vakiinnutetun rajapinnan läpi tällaiset alemmalla tasolla tapahtuvat muutokset eivät haittaisi, joten pitkällä tähtäimellä tämä on ehdottomasti kestävin tapa.

Osittain tällainen rajapinta on jo olemassakin; yhtiön tarjoama klinikanhallintaohjelmisto voi kommunikoida hoitoyksiköiden kanssa ethernetin kautta. Toistaiseksi rajapinnan kautta ei kuitenkaan ole saatavilla tarpeeksi paljon tietoa kaikkien diagnostiikkatarpeiden kattamiseen, mikä onkin pääasiallinen syy tämän diplomityön aloittamiselle.

3.3.4 Viestien tulkinta

Kokonaisuudessaan hammashoitoyksikön eri toimilaitteet käyttävät satoja erilaisia viestejä. Kaikkia näitä ei välttämättä tarvitse riittävän hyvän diagnostiikan aikaansaamiseksi käsitellä, mutta jo muutaman kymmenen viestin purkaminen tarkoittaa isohkoa työmäärää. Jos viestien sisältöjä tai tunnisteita myöhemmin muutetaan, täytyy ohjelmaa muuttaa vastaavasti. Ihanteellisessa tapauksessa viestien sisällöt kuvattaisiin jossakin erillisessä tiedostossa, jolloin muutokset peilautuisivat ohjelmaan automaattisesti. Tarkoitukseen käyttökelpoista formaattia ei kuitenkaan ole valmiiksi olemassa, joten on punnittava, kannattaako sellaista tämän projektin yhteydessä tehdä.

3.4 Toimintatapa

Ohjelmisto voidaan joko rajoittaa viestiliikenteen kuuntelemiseen tai siitä voidaan tehdä aktiivinen osa jolla voidaan myös ohjata järjestelmän toimintaa. Viestien lähetys mahdollistaa monipuolisemmat toiminnot, mutta on hankalammin toteutettavissa ja vaatii muutoksia itse hoitoyksikön toimintaan.

3.4.1 Viestiliikenteen passiivinen tarkkailu

Kuuntelemalla hoitoyksikön viestiliikennettä voidaan seurata merkittävää osaa sen toiminnasta hyvin tarkasti. Asioita, jotka tapahtuvat toimilaitteiden tai pääkortin ohjelmiston sisällä, mutta joista ei lähetetä ulospäin viestejä, ei päästä kuitenkaan seuraamaan.

Viestit voidaan kahteen päätyyppiin. Osa viesteistä on periodisia, eli toimilaitteet lähettävät niitä säännöllisen toistuvasti ilman eri pyyntöä. Näissä viesteissä kulkevia asioita on erityisen helppo seurata kuuntelemalla väylää. Toiset viestit taas esiintyvät vain tarvittaessa, esimerkiksi kun jokin tila muuttuu. Jälkimmäisessä tapauksessa voidaan joutua odottamaan pitkäänkin ennen kuin kyseiseen tietoon saadaan päivitys.

Eräs mahdollisuus olisi varata diagnostiikalle omia viestejä, joita lähettämällä ohjelmisto voisi pyytää tilannetietoja myös muutoin näkymättömistä asioista. Tällainen oma rajapinta voi kuitenkin olla resurssien haaskausta, sillä hoitoyksikön jatkokehityssuunnitelmissa on jo tiedossa integrointi ulkoiseen klinikanvalvontaohjelmistoon.

3.4.2 Hoitoyksikön aktiivinen etäkäyttö

Viestien lähettäminen suoraan toimilaitteille mahdollistaisi laajemman diagnostiikan kuin pelkkä viestien kuuntelu. Tällaista ominaisuutta ei kuitenkaan ole turvallista toteuttaa ilman tarkempaa suunnittelua, sillä laitteet tai pääkortti saattavat häiriintyä, jos järjestelmään syötetään viestejä ulkopuolelta. Aktiivisessa diagnostiikassa olisikin ehkä luoda oma rajapinta pääkortin ohjelmiston ja diagnostiikkaoh-

jelman välille, ja hoitaa tilannekyselyt sitä kautta. Mutta kuten aiemmin todettiin, kokonaan uuden rajapinnan luominen olisi suuri työrupeama ja edellyttäisi väkisinkin muutoksia hoitoyksikön muuhun ohjelmistoon.

3.5 Käyttöliittymä

Keskeinen tavoite tässä projektissa on tehdä hoitoyksiköstä huoltotoimenpiteiden kannalta helposti lähestyttävä. Tässä ei yksin riitä, että kaikki tarpeellinen tieto on nähtävillä, vaan ohjelmiston käytön helppoudella ja tietojen esitystavalla on myös suuri merkitys. Vaikka työkalu olisi miten tehokas, on se hyödytön, jos sitä ei osata käyttää.

Ihmisen ja tietokoneen välistä vuorovaikutusta (*human-computer interaction*, HCI) käsittelevää kirjallisuutta on paljon ja käyttöliittymien suunnittelu on merkittävä tutkimuskohde. Diagnostiikkaohjelmiston käyttöliittymä on suunniteltava riittävän helppokäyttöiseksi ja varmatoimiseksi. Varsinaisen kaupallisen tuotteen vaatimuksia ulkoasun suhteen ei välttämättä tarvitse noudattaa, sillä ohjelmisto on ainakin toistaiseksi tarkoitettu vain varsin rajattuun käyttöön.

Jotta ohjelman käyttöliittymän toteuttamisessa päästään parhaaseen mahdolliseen lopputulokseen, voi olla parasta suunnitella ohjelma käyttöliittymästä alkaen. Ensin tulee siis määritellä jokin lähtökohta käyttöliittymästä löytyville toiminnallisuuksille, sitten antaa prototyyppi testaajien käyttöön ja lopuksi korjata käyttöliittymää testauksessa havaittujen puutteiden mukaisesti. Näitä kierroksia täytyy ehkä tehdä useampiakin. [14]

Hyvä käyttöliittymä on sellainen, jolla käyttäjä saa haluamansa asiat tehtyä mahdollisimman nopeasti ja virheitä tapahtuu mahdollisimman vähän. Tiedot esitetään käyttäjälle selkeästi ja tarpeettomia tietoja ei esitetä. Tietojen tulee olla mahdollisimman helposti tulkittavissa väärinkäsitysten ehkäisemiseksi. [15]

Suuri määrä numeerista tietoa on useimmiten havainnollisempaa esittää jonkinlaisena graafisena esityksenä. Useimmiten parhaaseen lopputulokseen ei kuitenkaan päästä lataamalla kerätty tieto taulukkolaskentaohjelmaan ja luomalla nopeasti kuvaaja oletusasetuksilla, vaan esitystapaan kannattaa kiinnittää huomiota. Kirja *The Visual Display of Quantitative Information* on arvostettu kirja tiedon visualisoinnin tutkimuksen alalla. Teoksessa kirjoittaja Tufte esittää graafisen esityksen suunnittelusta seuraavanlaisen yhteenvedon: [16, s. 105]

> *Above all else show the data.*
>
> *Maximize the data-ink ratio.*
>
> *Erase non-data-ink.*
>
> *Erase redundant data-ink.*
>
> *Revise and edit.*

Vapaasti suomennettuna ensimmäinen kohta tarkoittaa, että kuva on suunniteltava esitettävää tietoa unohtamatta; vaikka kuva olisi miten hienosti toteutettu, se on arvoton, jos se ei välitä lukijalle haluttua tietoa.

Termi *data-ink ratio* tarkoittaa varsinaiseen tietoon käytetyn painomusteen suhdetta koko kuvaan käytettyyn musteeseen. Mitä suurempi tämä suhde on, sen tehokkaammin kuva yleensä onnistuu siirtämään tietoa lukijalle. Kahdessa seuraavassa kohdassa ohjeistetaankin poistamaan kuvasta tietoon liittymättömiä tai saman tiedon useampaan kertaan esittäviä elementtejä.

Viimeinen ohje kehottaa lopuksi tarkastelemaan lopputulosta ja tekemään tarvittaessa lisää muutoksia. Hoitoyksikkö sisältää paljon asioita, joiden esittämiseen jonkinlainen kuva tai kuvaaja on todennäköisesti toimivin vaihtoehto, joten näille periaatteille löytynee työssä käyttöä.

4 Tulokset

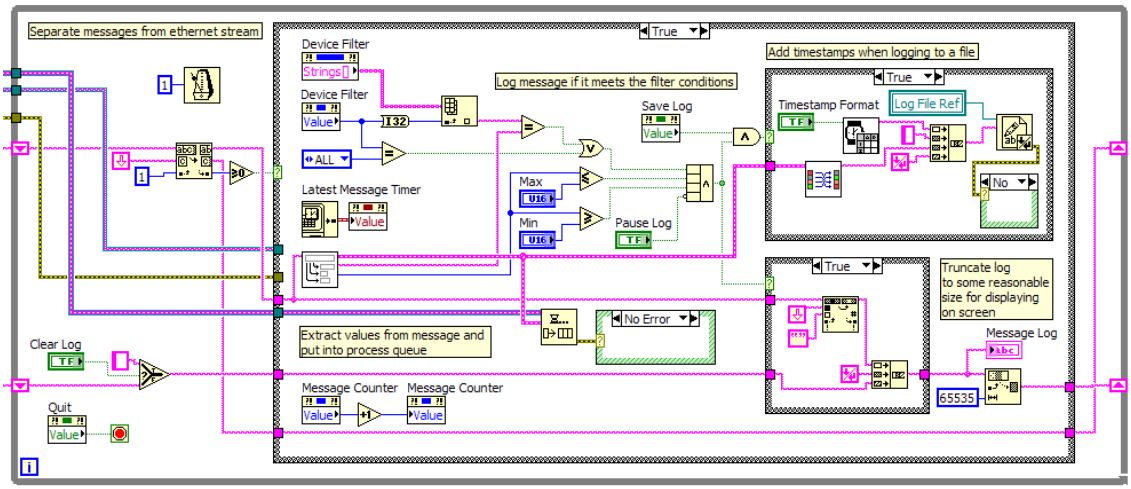
Työn tässä osassa esitellään kokeellisen osan tulokset.

4.1 Ohjelmiston rakenne

Edellä käsiteltyjen asioiden puitteissa toteutettavaksi ratkaisuksi muodostui LabVIEW-ohjelmisto, joka tulkitsee hoitoyksikön pääkortin ethernetiin välittämiä CAN-viestejä. Ohjelmisto pyrkii vastaamaan tuotannon ja huoltohenkilökunnan esittämiin toiveisiin. Toteutuksessa on tähdätty yksinkertaisuuteen ja modulaarisuuteen, jotta muutkin voivat hoitaa ohjelmiston jatkokehitystä.

4.1.1 Yleistä

Ohjelman perusrakenne muistuttaa mitä hyvänsä sovellusta. Ensimmäiseksi alustetaan tarpeelliset muuttujat, luetaan konfiguraatiotiedostot ja niin edelleen. Kun alustus on onnistuneesti suoritettu, ohjelma voi siirtyä toimintavaiheeseen odottelemaan käyttäjän syötettä. Kun käyttäjä sulkee ohjelman, suoritetaan ethernet-yhteyden asianmukainen katkaisu ja muut vastaavat lopputoimenpiteet. Virhetilanteiden käsittely on pyritty hoitamaan mahdollisimman siististi läpi sovelluksen.



Kuva 3: Esimerkki LabVIEW-ohjelmasta

Kuvassa 3 on esimerkki LabVIEW'lla toteutetusta silmukasta. Silmukkaan tuodaan sisälle alkuasetukset sen ulkopuolelta ja käsittely etenee sitten järjestyksessä vasemmalta oikealle. Kun kaikki silmukan sisällä olevat toiminnot on suoritettu loppuun,

aloitetaan alusta. Kuvan oikealla puolella oleva sisempi laatikko kuvaa ehtorakennetta, jonka sisältämät toiminnot suoritetaan vain jos sen heräte (vihreä kysymysmerkki laatikon vasemmassa reunassa) on tosi, esimerkkitapauksessa siis jos käsiteltävän merkkijonon pituus on suurempi kuin 0. Sisemmillä ehtorakenteilla on nähtävissä monimutkaisemmat herätteet. Silmukan suoritus lopetetaan, kun vasemmalla alhaalla näkyvän *Quit*-muuttujan arvo muuttuu todeksi.

4.1.2 Ethernet-yhteyden hallinta

Jo heti projektin alkuvaiheessa toteutin LabVIEW'lla yksinkertaisen sovelluksen, joka käytti tuotekehityksestä jo löytyviä CAN-sovittimia viestien lukemiseen ja lähettämiseen yksittäiselle toimilaitteelle tai hoitoyksikön väylälle. Sovellus osaa myös selvittää omatoimisesti toimilaitteille jaetut osoiteavaruudet, jos se on kytkettynä hoitoyksikköön heti käynnistysvaiheessa.

Pian huomio kääntyi kuitenkin ethernetiin, sillä sitä hyödyntämällä vältytään erillisten CAN-sovittimien käytöltä ja sitä kautta kaikenlaisilta ajuri- ja yhteensopivuusongelmilta. CAN-sovittimet tietysti myös maksavat, kun taas ethernet-liitäntä löytyy käytännössä kaikista tietokoneista. Sisäisesti ohjelmisto on kuitenkin suunniteltu siten, että viestien käsittelyssä ei ole oleellisia eroja ja viestien luku suoraan CAN-väylältä on melko nopeasti toteutettavissa, jos se osoittautuu joskus tarpeelliseksi. Kokeiluvaiheessa syntynyt CAN-väylään liittyvä ohjelmakoodi on sittemmin otettu käyttöön testiautomaation tarpeisiin, sillä ethernetin käyttö ei ole niissä olosuhteissa mahdollista.

Ethernet-toteutuksessa menetettiin toistaiseksi mahdollisuus lähettää viestejä hoitoyksikölle päin, koska pääkortti ei vielä välitä liikennettä toiseen suuntaan. Käytännössä suurin osa diagnostiikasta on joka tapauksessa hoidettavissa kuuntelemalla viestiliikennettä, joten asia ei lyhyellä tähtäimellä rajoita ohjelmiston kehitystä liialti. Myös mahdollisuus kommunikoida yksittäisen toimilaitteen kanssa hoitoyksiköstä erillään poistui päätöksen myötä, mutta tarvetta tällaiselle on tuotekehitysosaston ulkopuolella harvoin ja tuotekehityksellä on joka tapauksessa ennestään tarkoitukseen sopivia työkaluja.

4.1.3 Viestien luku verkosta

Ohjelmiston ytimenä ajetaan kolmea rinnakkaista silmukkaa. Silmukoista ensimmäinen muodostaa tilakoneen, joka pitää yllä ethernet-yhteyttä hoitoyksikköön ja lukee verkosta jatkuvasti tietoa. Yhteys muodostetaan ip-osoitteen ja portin perusteella, jolloin seurattava laite voi fyysisesti sijaita kaukanakin. LabVIEW'sta löytyy suoraan käyttökelpoisia ethernet-toimintoja joiden avulla yhteyden hallinta onnistuu vähällä vaivalla.

ACCU-kortin ohjelmisto muuntaa kaikki väylän CAN-viestit ihmisen luettavaan tekstimuotoon ja lähettää ne ethernet-paketteina verkkoon. Tietojen lukeminen tapahtuu pakettimuodossa siten, että tavuja luetaan joko tietty määrä (oletusarvoi-

sesti 2048 tavua) kerrallaan tai kunnes ennalta päätetty määräaika (oletusarvoisesti 100 ms) täyttyy. Jos silmukassa havaitaan että edellisestä viestistä on kulunut liian pitkään (oletusarvoisesti 10 s), tilakone siirtyy lukutilasta takaisin yhdistämistilaan ja pyrkii muodostamaan yhteyden uudelleen. Tällä varmistetaan, että pidempiaikaisessakin seurannassa saadaan viestit talteen, vaikka hoitoyksikkö välillä sammutettaisiin.

Tässä ensimmäisessä vaiheessa luettuja tietoja ei sen ihmeemmin käsitellä. Tiedot siirretään odottamaan LabVIEW'n *queue*- eli jonorakenteeseen, josta ne luetaan seuraavaan silmukkaan.

4.1.4 Viestien purku

Viestejä purkava silmukka lukee merkkejä ensimmäisen silmukan täyttämästä jonosta. Merkkivirrasta etsitään rivinvaihtoja, joita valitussa esitysmuodossa käytetään erottamaan viestit toisistaan. Rivinvaihdon löytyessä sitä edeltävät merkit yritetään tulkita CAN-viestiksi perinteisillä *sscanf*-tyyppisillä C-kielestä tutuilla merkkijonofunktioilla. Viestiin kuuluva toimilaitteen nimi muunnetaan samalla diagnostiikkaohjelman oman nimeämiskäytännön mukaiseksi. Muunnos tehdään lähinnä sen vuoksi, että ohjelman käsiteltäväksi voitaisiin tarvittaessa tässä välissä syöttää viestejä myös muista lähteistä, kuten suoraan CAN-sovittimelta. Muista lähteistä tulevat viestit eivät todennäköisesti sisältäisi toimilaitteen nimeä samalla tavalla, jos lainkaan, joten asia on hyvä huomioida heti alkuvaiheessa.

Samassa silmukassa suoritetaan myös viestien esittäminen käyttäjälle sekä niiden tallennus lokiin. Useimmiten viestejä ei toki tarkastella sellaisinaan - ohjelman tarkoitushan on juuri esittää tiedot selvässä muodossa - mutta ominaisuus on silti hyvä olla olemassa erikoistapauksia varten. Viestejä voi myös suodattaa toimilaitteen nimen ja viestin numeron perusteella, jos vain jokin yksittäinen laite tai viesti kiinnostaa.

Viestien tallennus lokiin tapahtuu näytöllä esitettävän näkymän perusteella; valitut suodatusasetukset pätevät myös tallennettaviin viesteihin. Lisäksi lokiin lisätään kunkin viestin kohdalle aikaleima joko luettavassa tai Excel-taulukkolaskennan käyttämässä muodossa. Aikaleiman tarkkuus ei vastaa tarkkuudeltaan viestien ajoitusta, sillä viestien välinen aika muuttuu matkalla ethernetin läpi. Aikaleiman avulla voi kuitenkin paikantaa esimerkiksi jonakin tiettynä ajanhetkenä tapahtuneen vian summittaisen sijainnin lokissa. Lokitiedoston nimi muodostetaan päivämäärän perusteella.

Kuvan 3 esimerkki on juuri tästä silmukasta.

4.1.5 Viestien sisällön tulkinta ja esitys

Kolmas silmukka lukee jälleen viestit jonosta ja jakaa ne oikeille toimilaitemoduuleille. Tässä vaiheessa voidaan myös päättää, halutaanko esimerkiksi toistuvista viesteistä käsitellä kuorman vähentämiseksi vain osa tai tehdä joitakin muita suodatuk-

sia. Silmukassa päivitetään myös kunkin toimilaitteen osalta ajastinta, joka kertoo viimeisimmän viestin saapumishetken.

Kullekin toimilaitteelle on tehty oma moduuli, jolle lähetetään käsiteltäviksi vain sille kuuluvat viestit. Moduulin sisällä viesti ohjataan tunnistenumeronsa mukaisesti oikealle käsittelijälle, jossa viestin tiedot puretaan ja esitetään käyttäjälle. Kaikille viesteille ei ole vielä käsittelijöitä, vaan niitä on rakennettu sitä mukaa kun tarvetta uudelle diagnostiikalle on ilmennyt. Yksittäisten toimilaitemoduulien toiminnasta kerrotaan tarkemmin seuraavissa osioissa.

Kukin moduuli koostuu pääasiassa muutamasta sisäisestä valintarakenteesta. Uloin rakenne tarkistaa varmuuden vuoksi että käsiteltävä viesti varmasti kuuluu oikealle toimilaitteelle. Jos laite on oikea, siirretään viesti toiseen valintarakenteeseen, joka aktivoituu viestin tunnistenumeron perusteella. Tämän jälkeen voi olla vielä kolmas rakenne, jos kyseessä on dynaaminen viesti. Kun tiedetään mistä viestistä on kysymys, voidaan sisältö avata ja käsitellä oikealla tavalla. Jotkin asiat esitetään suoraan käyttäjälle esimerkiksi numerona, toiset saattavat riippua jostakin aiemmasta tilasta tai viestistä.

Moduulit on suunniteltu siten, että ne aktivoituvat vain uuden viestin saapuessa, eikä niissä siis pyöri jatkuvasti omaa silmukkaa esimerkiksi käyttöliittymän tapahtumien seurantaa varten. Tämä toteutus pitää suorittimen kuormituksen alhaisena, mutta voi tietyissä tapauksissa aiheuttaa käyttäjän näkökulmasta viiveitä, kun esimerkiksi napinpainallukseen reagoidaan vasta seuraavan viestin jälleen aktivoidessa moduulin. Käytännössä tämän ei pitäisi olla merkittävä haitta oikeassa käytössä, koska seurattava laite on tyypillisesti aktiivisena ja uusia viestejä saapuu tasaiseen tahtiin. Jos asia kuitenkin myöhemmin koetaan ongelmaksi, voidaan moduuleille lähettää pääohjelmasta tyhjiä viestejä, jotka eivät sisällä varsinaista tietoa, mutta joiden saapuminen päivittää käyttöliittymän tilan. Jos tämäkään ei riitä, voidaan tietysti siirtyä kunkin moduulin käyttöliittymän tapahtumien täydelliseen seurantaan.

4.1.6 Käyttöliittymän hallinta

Käyttöliittymän painalluksia ja muita tapahtumia seurataan LabVIEW'n *event*- eli tapahtumarakenteessa. Rakenteeseen voi valita haluamansa asiat, esimerkiksi yksittäisen nappien painalluksen, hiiren kuljettamisen tietyn alueen päälle, ikkunoiden sulkemisen tai muuta vastaavaa. Kullekin tapahtumalle voi määritellä haluamansa toimintaketjun; esimerkiksi käyttäjän painaessa jotakin toimilaitetta vastaavaa nappia voidaan avata kyseinen ikkuna. Tapahtumarakenne tekee käyttöliittymän hallinnasta joustavaa ja tehokasta.

4.1.7 Laajennettavuus

Tämän diplomityön puitteissa ei ollut mahdollista toteuttaa kaikkien hoitoyksikön ominaisuuksien seurantaa. Ohjelmisto onkin jaettu toimilaitteittain moduuleihin,

joita voidaan jatkossa kehittää ja lisätä. Kukin toimilaitemoduuli on itsenäinen kokonaisuus, jolle voidaan ohjata viestejä muutakin kautta tai käyttää moduulia osana jotakin toista sovellusta.

4.2 Toteutetut ominaisuudet

Rakensin ohjelmistoon ominaisuuksia siinä tärkeysjärjestyksessä mikä kerättyjen mielipiteiden mukaan vaikutti oikeimmalta. Joitain erityisen työläitä ominaisuuksia on lykätty nopeammin toteutettavien kustannuksella.

4.2.1 Yhteys hoitoyksikköön

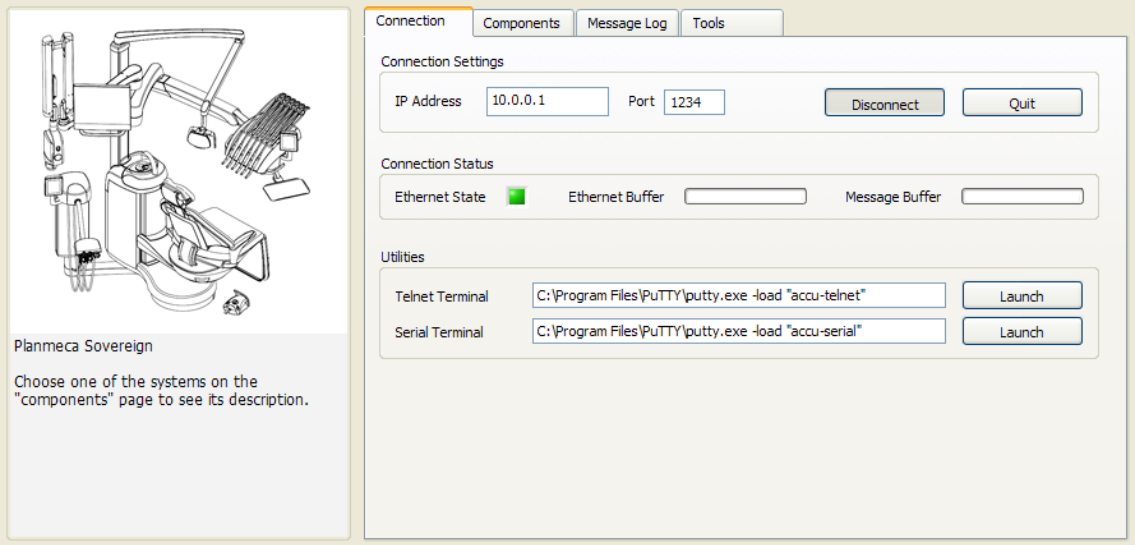
Aluksi oli järjestettävä yhteys tietokoneen ja hammashoitoyksikön välille ethernet-väylää pitkin. LabVIEW tarjosi valmiita funktioita yhteyden muodostamiseen ja alustava tilakone yhteyden muodostamiseen ja tietojen lukuun väylältä olikin pian koossa. Käyttäjälle ohjelman tästä osasta näkyvät ip-osoitteen ja portin valintakentät sekä yhdistämispainike (*connect*) kuvan 4 osoittamalla tavalla. Lisäksi yhteyden tilasta kertovat alapuolelta löytyvä merkkivalo sekä jonojen puskureiden täyttöasteita kuvaavat palkkimittarit.

Koska ajatuksena oli että ohjelma soveltuisi myös pidempiaikaiseen käytönseurantaan ja voisi kerätä viestilokia esimerkiksi viikon ajalta, oli tilakone suunniteltava siten, että se osaa muodostaa yhteyden uudelleen vaikka hammashoitoyksikkö välillä sammutettaisiin. Kuten aiemmin käsiteltiin, tämä on ohjelmassa järjestetty niin, että jos uusia viestejä ei havaita kymmenen sekunnin aikana, verkkoyhteys suljetaan ja tilalle yritetään muodostaa uusi. Yhteyden muodostamista yritetään kunnes siinä onnistutaan tai käyttäjä katkaisee yhteyden kokonaan. *Ethernet state* -merkkivalo on yhteyden ollessa aktiivinen vihreä, uudelleenyritysten aikana keltainen ja muulloin musta.

Eräs toive oli mahdollisuus laukaista suoraan uudesta diagnostiikkaohjelmistosta puhdas sarja- tai telnet-yhteys hoitoyksikön pääkortille. Tätä silmälläpitäen yhteysnäkymään on lisätty paikat kahdelle vapaavalintaiselle komennolle jotka voi suorittaa napin painalluksella. Luomalla käytettävään pääteohjelmistoon valmiiksi oikeat yhteysprofiilit yhteyden muodostaminen onnistuu siis helposti. Integroidumpi ratkaisu olisi tietysti rakentaa kyseiset pääteohjelmistot suoraan diagnostiikkaohjelmistoon, mutta sellaiseen ei ollut resursseja.

4.2.2 Viestien seuraaminen ja tallentaminen

Vaikka tämän ohjelmiston tarkoitus onkin esittää hoitoyksikön viestiliikenne valmiiksi tulkittuna, voi joissain tapauksissa olla hyödyllistä päästä seuraamaan suoraan mitä väylällä tapahtuu. Tätä varten on kuvan 5 mukainen näkymä, jossa kaikki vastaanotetut viestit ovat nähtävillä. Viestejä on tarvittaessa mahdollista seuloa toimilaitteen ja/tai viestin tunnisteen mukaan.



Kuva 4: Ohjelman yhteysasetukset

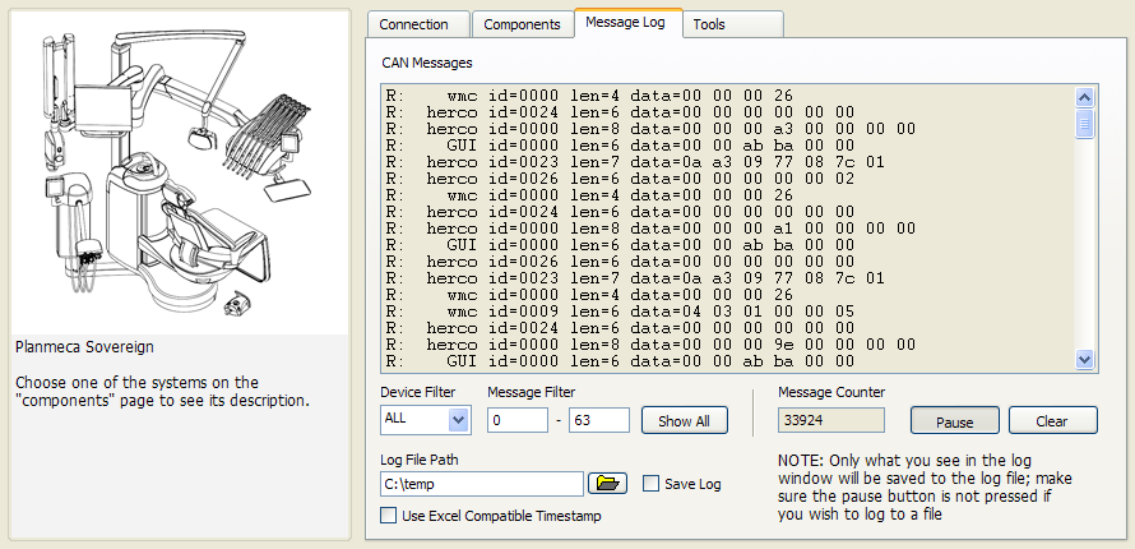
Kaikki viesti-ikkunassa näkyvät viestit on samalla mahdollista tallentaa tiedostoon. Tiedoston nimi muodostuu automaattisesti ajankohdan perusteella, jotta vanhoja lokeja ei ylikirjoiteta eikä toisaalta pääse syntymään yhtä valtavan pitkää lokitiedostoa. Viesteihin voi tallentaa aikaleiman joko ihmisten luettavassa muodossa tai Excel-taulukkolaskentaohjelmiston ymmärtämässä muodossa.

4.2.3 Toimilaitenäkymät

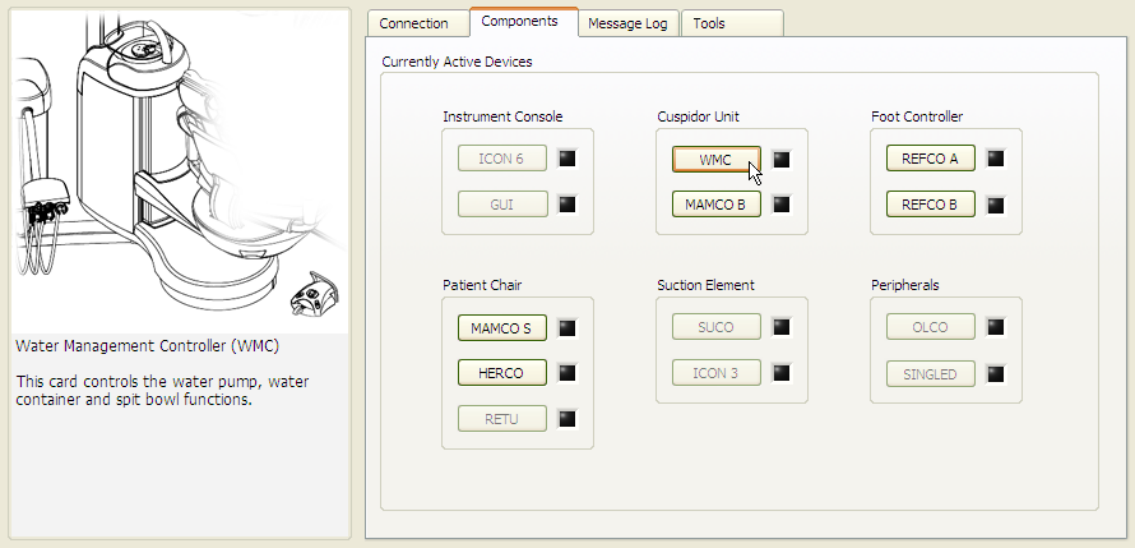
Kaikkien toimilaitteiden näkymien kerääminen samaan ikkunaan vaikutti liian sekavalta, joten päätin antaa kullekin toimilaitteelle oman ikkunan. Ratkaisu ei ole täydellinen, sillä kovin monen ikkunan avaaminen yhtäaikaa voi täyttää tietokoneen näyttötilan nopeasti. Toisaalta kaikkien tietojen sullominen yhteen ikkunaan välilehtien taakse tekisi useamman asian yhtäaikaisesta seuraamisesta vaivalloista. Jonkinlainen yhdistelmä, esimerkiksi irroitettavat ja uudelleenjärjesteltävissä olevat paneelit, olisi monipuolisin ratkaisu, mutta tällaisen toteuttaminen LabVIEW'lla vaikutti lyhyen arvion jälkeen liian työläältä.

Toimilaitevalintanäkymä esitetään kuvassa 6. Kullakin toimilaitteella on oma painikkeensa sekä merkkivalo, joka ilmaisee väylällä kulkevan kyseisen laitteen viestejä. Toimilaitteiden, joille ei vielä ole näkymää, painikkeet ovat himmennettyjä eikä niitä voi käyttää.

Käytettävyyteen liittyen oli esitetty toive interaktiivisesta aputoiminnosta. Koska kaikki ohjelman aiotut käyttäjät eivät välttämättä ole täysin perillä eri piirikorttien nimistä ja toiminnoista, haluttiin lisätä jonkinlaista pehmeyttä pelkkien mi-



Kuva 5: Viestilokin asetukset



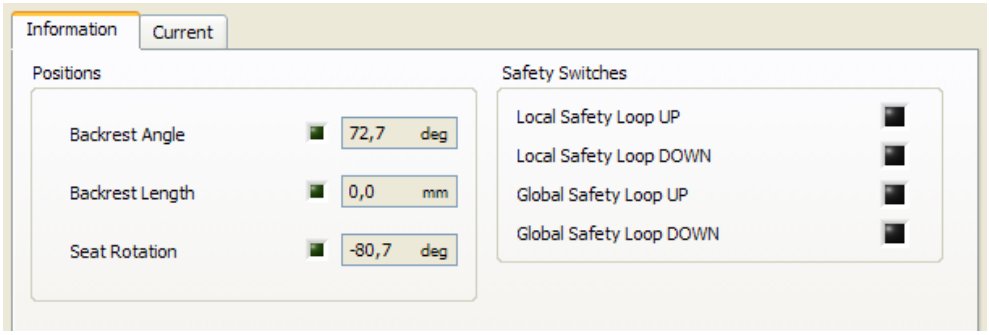
Kuva 6: Toimilaitenäkymien valinta

täänsanomattomien kirjainlyhennetunnisteiden ohelle. Asiaa on pyritty helpottamaan kokoamalla toimilaitekortit loogisiin ryhmiin ja lisäämällä ikkunaan apukuva ja -teksti, joka vaihtuu automaattisesti käyttäjän viedessä osoitin painikkeen päälle. Kuva osoittaa karkeasti kortin konkreettisen sijaintipaikan ja teksti kertoo mitä omi-

naisuuksia kyseinen osa hallinnoi. Samaan apunäkymään on tarvittaessa mahdollista lisätä vastaavia tiivistelmiä ohjelman muistakin toiminnoista. Tämän aputoiminnon toteuttamisessa käytettiin pohjana yhtiön käytettävyysasiantuntijan aiemmin suunnittelemia luonnoksia.

4.2.4 MAMCO

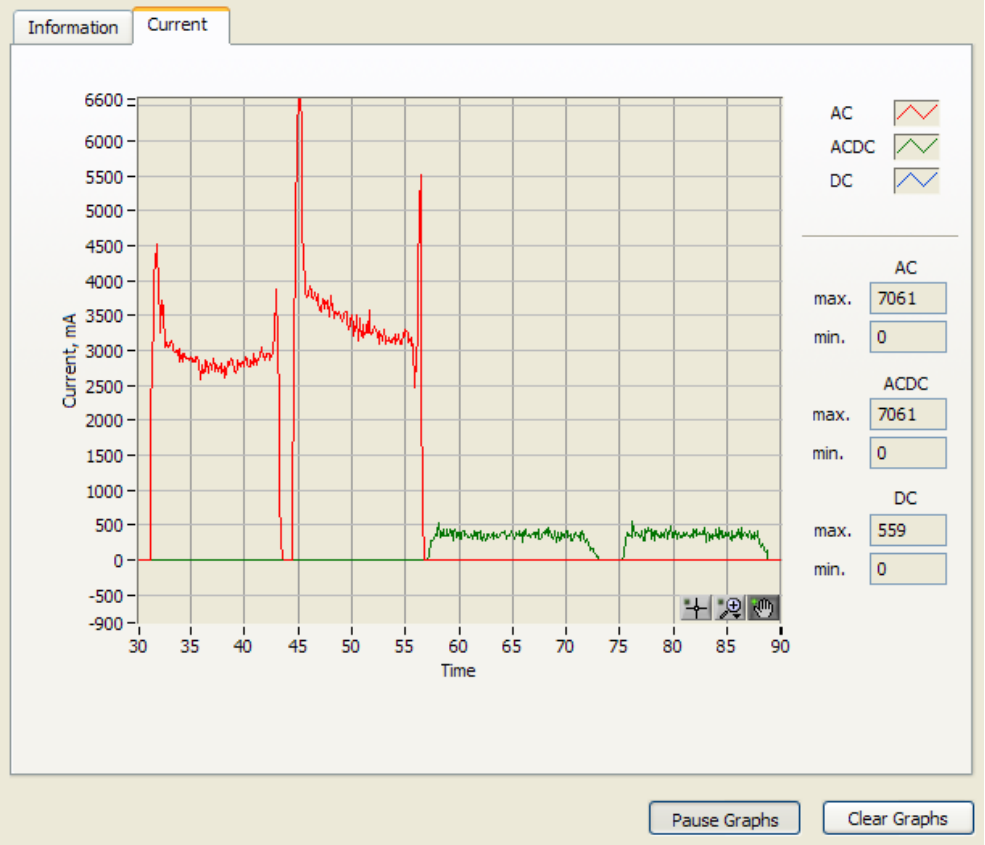
MAMCO-korttien osalta toivotuimmat ominaisuudet liittyivät moottorien asentoanturien kalibrointiin ja moottorien virranmittauksiin. Asentoanturien kalibrointia ei ole mahdollista varsinaisesti suorittaa ohjelmasta käsin, mutta on aivan mahdollista esittää visuaalisesti, kun anturi on säädetty oikeaan kalibrointikohtaan. Tätä tarkoitusta varten ohjelma seuraa väylällä kulkevia viestejä ja esittää kunkin anturin lukeman näytöllä, kuten kuvasta 7 ilmenee. Kaikkien anturien tulokset esitetään aina joko millimetreinä tai asteina liikeradan tyypistä riippuen. Kalibrointikohtaa merkitään pienellä vihreällä merkkivalolla asentotiedon vieressä.



Kuva 7: MAMCO-korttien yleisnäkymä

Samassa näkymässä näytetään myös MAMCO:n turvasilmukoiden tila. Hoitoyksikössä on useita turvakytkimiä, jotka aktivoituvat esimerkiksi sellaisissa tilanteissa joissa jokin estää potilastuolin liikkeen. Jos turvasilmukka katkeaa mistään kohdasta, moottoreiden ajo lopetetaan välittömästi ja käyttäjälle esitetään virheilmoitus. Turvasilmukoiden tarkastamiseen tuotannossa toivottiin testisekvenssiä, jonka avulla kytkimien toiminta voidaan käydä läpi. Tällaista testiä ei ole vielä laadittu sen tarkemmin, mutta kun käsittelijä näille viesteille on olemassa, ei testin toteuttaminen ole vaikeaa.

Moottorien virranmittausta varten on kuvaaja, jolle piirtyy kunkin moottorin virrankulutus, jos mittaustoiminto on käynnissä. Kuvaaja tallentaa virtaprofiilia viimeisen kymmenen minuutin ajalta sekä minimi- ja maksimitulokset kullekin moottorille koko seuranta-ajalta. Maksimiarvojen seuranta lisättiin pidempikestoisia testejä silmälläpitäen, jotta mahdolliset virranylitykset eivät jää huomaamatta, vaikka niitä ei kuvaajassa enää näkyisikään. MAMCO:n virranmittausnäkymä on esitetty kuvassa 8.



Kuva 8: MAMCO-korttien virranmittausnäkymä

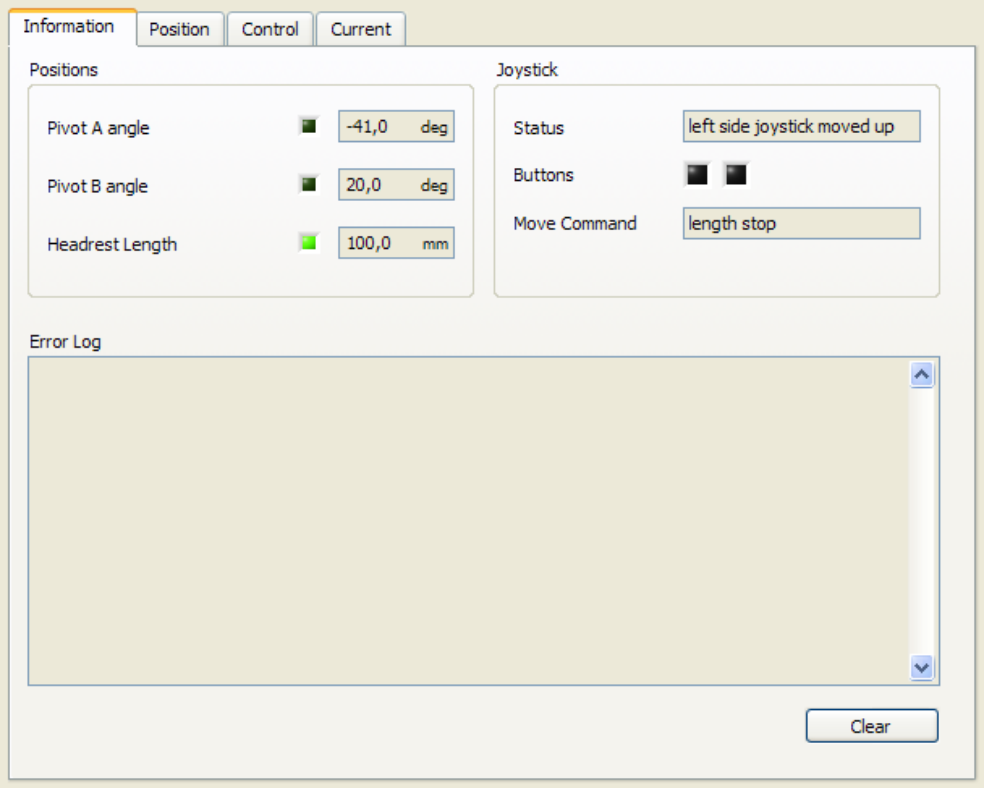
Virranmittaus on toimenpide, joka ei ole oletusarvoisesti käynnissä, vaan se on erikseen aktivoitava hoitoyksiköstä. Tämän tekeminen suoraan diagnostiikkaohjelmistosta olisi optimaalista, mutta yksisuuntaisen tietoliikenteen vuoksi se ei ollut vielä mahdollista.

4.2.5 HERCO

Potilastuolin moottoroitua niskatukea ohjaavan HERCO-kortin toiminta muistuttaa MAMCO-kortteja, joten diagnostiikkanäkymissä on pyritty yhdenmukaiseen ulkoasuun. Koska olen itse tehnyt myös HERCO:n ohjelmistoa, tarjoutui tässä samalla hyvä tilaisuus lisätä laitteeseen uusia diagnostiikkaviestejä ja tehdä niille tarpeelliset näkymät. Viestit on käynnistettävä hoitoyksiköstä erikseen, jotta ne eivät turhaan kuormita CAN-väylää tavallisen käytön yhteydessä.

MAMCO:jen tavoin HERCO:lla voi ohjata kolmea moottoria. Kunkin moottorin asentotieto ilmoitetaan vastaavalla tavalla asteina tai millimetreinä ja pieni merkkivalo ilmoittaa kun anturi on kalibrointipisteessä. Näiden ohella yleisnäkymässä, joka on esitetty kuvassa 9, näytetään lisäksi niskatuen ohjaintikkujen asennot sekä

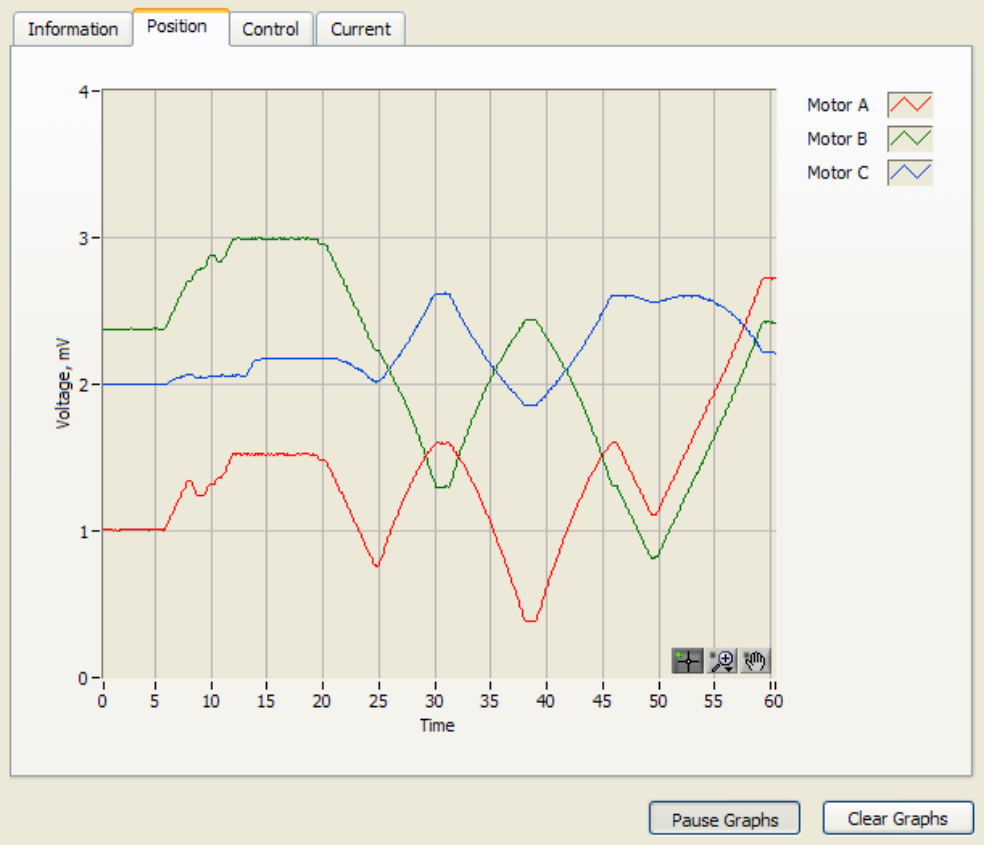
loki tapahtuneista virheistä. Lokille on käyttöä, sillä niskatuen virheitä ei esitetä kosketusnäytöllä.



Kuva 9: HERCO-kortin yleisnäkymä

Kuten kohdassa 2.4.1 aiemmin esitettiin, oli niskatuen toiminnassa havaittu välillä ongelmia. Jotta asentoanturien toiminnasta voitaisiin varmistua, lisäsin HERCO-kortin ohjelmistoon mahdollisuuden lähettää asentoanturien lukemia tasaiseen tahtiin. Nappaamalla nämä viestit kiinni diagnostiikkaohjelmassa ja piirtämällä niistä kuvaaja ajan suhteen tuli mahdolliseksi havaita lyhytaikaisetkin katkokset anturien toiminnassa. Kuvaajaan, joka on esitetty kuvassa 10, tallentuu mittaustuloksia viimeisen kymmenen minuutin ajalta.

Kuvassa 11 on kuvaaja, joka saatiin tulokseksi rikkonaiseksi epäillystä niskatuesta. On selvästi nähtävissä, miten asentoanturien signaali katoaa yhtäkkisesti muutamaksi sekunniksi kesken liikkeen. Ongelman syyksi paljastui lopulta rikkonainen kaapeli, joka tietyssä asennossa aiheutti mittauspiiriin oikosulun tai kelluvan jännitteen, mikä sai HERCO-kortin ohjelmiston pysäyttämään moottorit. Tällainen vika olisi sinänsä aivan mahdollista paikantaa yleismittarilla, mutta käytännössä johtoja ei pääse suojapeltien alta vähällä mittaamaan ja lisäksi ongelma tuli ja meni niskatuen asennosta riippuen. Tällaisessa tapauksessa CAN-viestien perusteella tehtävä diagnostiikka palveli siis oivasti tarkoitusta.



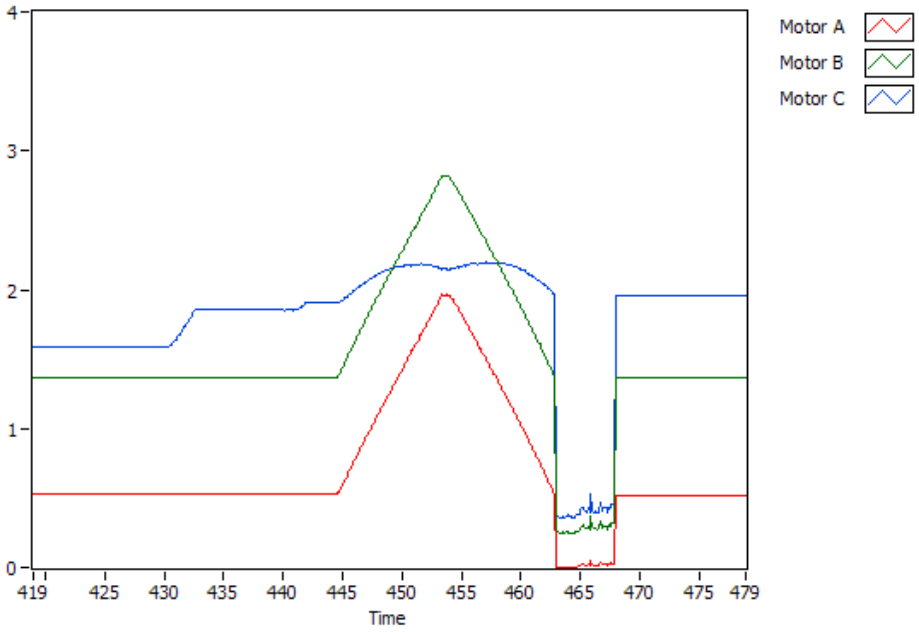
Kuva 10: HERCO-kortin asennonmittausnäkymä

Asentoanturien lisäksi tein mittausnäkymät myös moottorien ohjausjännitteille ja virroille. Molempien perusteella on mahdollista diagnosoida erinäisiä ongelmia, mutta niistä on apua myös tuotekehitystöissä esimerkiksi moottorin säätimien ohjausparametrien asettelussa. Toki tässä tapauksessa säätimet olivat jo enemmän tai vähemmän valmiit, mutta jos esimerkiksi jonkin niskatuessa käytettävän moottorin malli joskus vaihtuu, on näiden kuvaajien avulla helppoa asetella säädinten toiminta entistä vastaavaksi.

4.2.6 REFCO

Hoitoyksikön jalkaohjaimella käyttäjä voi ohjata instrumentteja tai aktivoida kosketusnäytöllä näkyviä toimintoja. Jalkaohjaimen poljinta voi liikuttaa vaaka- ja pystysuunnassa ja sen asento tunnistetaan kapasitiivisella anturilla. Tällaisen anturin kalibrointi on perinteistä potentiometriä tai mikrokytkintä vaativampi toimenpide ja sen yksityiskohdat ovat aiheuttaneet aiheeseen tutustumattomissa hämmennystä, mikä on asia johon toivottiin parannusta.

Kuvassa 14 esitetään jalkaohjaimen yleistietonäkymä. Näkymästä on mahdollista



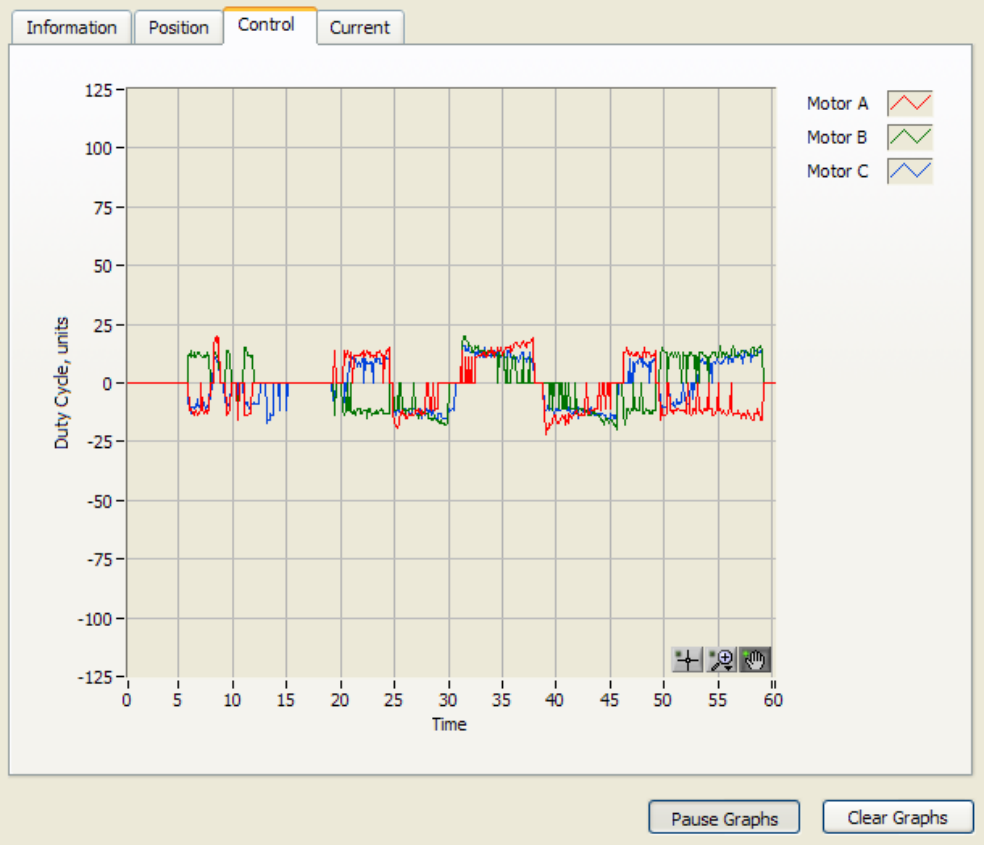
Kuva 11: Niskatuen asentoantureissa havaittu yhteyskatkos

varmistaa että poljin saavuttaa ääriasennoissaan maksimiarvot ja että kaikki jalkaohjaimen painikkeet toimivat oikein. Lisäksi merkkivaloilla ilmaistaan, onko jalkaohjain tällä hetkellä kalibrointitilassa, onko poljin tällä hetkellä tallennettujen kalibrointirajojen sisällä sekä joitain muita yksityiskohtia.

Kalibrointinäkymässä, kuvassa 15, pääsee seuraamaan kalibroinnin tarkkoja vaiheita. Polkimen kalibrointi tapahtuu kuljettamalla se kaikkien ääriasentojen läpi, jolloin toimilaitteen ohjelmisto voi laskea tarpeelliset raja-arvot. Näitä kalibrointikohtia polkimella on kuusi kappaletta ja diagnostiikkanäkymässä esitetään missä vaiheessa kulloinkin ollaan. Myös kapasitiivisen anturin elektrodien mittaustulokset esitetään, koska näitä tietoja saatetaan tarvita jalkaohjaimen kokoonpanon yhteydessä.

Kun kaikki asennot on käyty läpi jalkaohjain lähettää palautteena viestin, joka sisältää tiedon siitä hyväksyttiinkö kalibrointiyritys. Epäonnistuneesta yrityksestä palautetaan virhekoodi, jonka perusteella voidaan päätellä mikä meni vikaan. Tässä olisi hyvä tilaisuus soveltaa myös interaktiivista apunäkymää, jossa voitaisiin esittää ohjeita ongelman korjaamiseksi. Tätä toimintoa ei kuitenkaan vielä ole toteutettu.

Jalkaohjaimia on kahdenlaisia, toinen on yhdistetty kaapelilla suoraan hoitoyksikköön ja toinen toimii akkujen varassa langattomasti. Molemmat käyttävät lähes identtistä viestiprotokollaa, mutta langattoman ohjaimen tapauksessa viesteissä kulkee tietoa myös akun varaustilasta, laturista ja muista vain sitä koskevista asioista. Akun ja laturin tilat esitetään omalla välilehdellään, joka on valittavissa vain silloin kun käytössä on langaton ohjain. Muilta osin viestien sisältö on täsmälleen samanlaista, joten ohjaimille ei ole tehty erillisiä näkymiä vaan molempien viestit ohjataan



Kuva 12: HERCO-kortin jännitteenohjausnäkymä

samalle käsitteljälle. Ratkaisu ei aiheuta sekaannusta, sillä hoitoyksikköä käytetään aina yhdellä jalkaohjaimella kerrallaan.

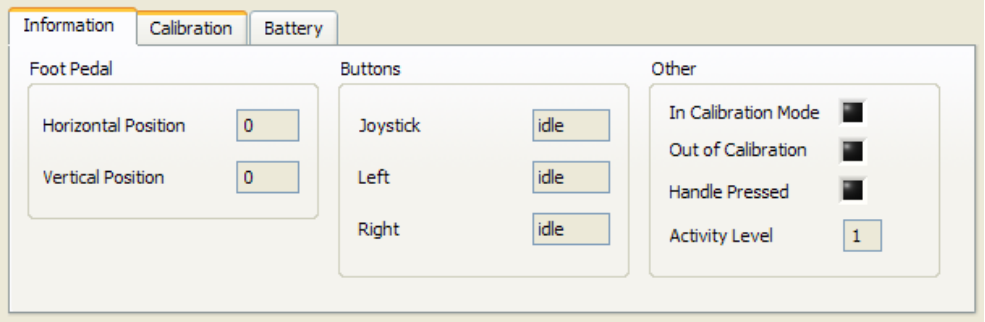
Aiemmin, kohdassa 2.4.1, oli maininta langattoman jalkaohjaimen akkujen kestoon liittyvästä epäselvyydestä. Asiaa tutkittiin diagnostiikkaohjelmiston avulla seuraamalla erään hoitoyksikön käyttöä hammasklinikalla. Kerättyjen lokien perusteella voitiin nähdä, että hammaslääkäri käyttää jalkaohjainta aktiivisemmin kuin mitä tuotekehityksessä oli käsitetty. Jalkaohjaimen virransäästöominaisuuksia parannettiin havaintojen pohjalta odotuksia vastaaviksi.

4.2.7 WMC

WMC-kortin toimiala on edellisiä laitteita monipuolisempi. Se hallinnoi ennen kaikkea hoitoyksikön lukuisia venttiileitä ja releitä, mutta ohjaa myös vesipumppua, paineantureita, sylkymaljan kääntömoottoria ja vesihanan ultraääniantureita. Toivotuimpia ominaisuuksia WMC:n osalta olivat venttiilien ja paineanturien tilat.

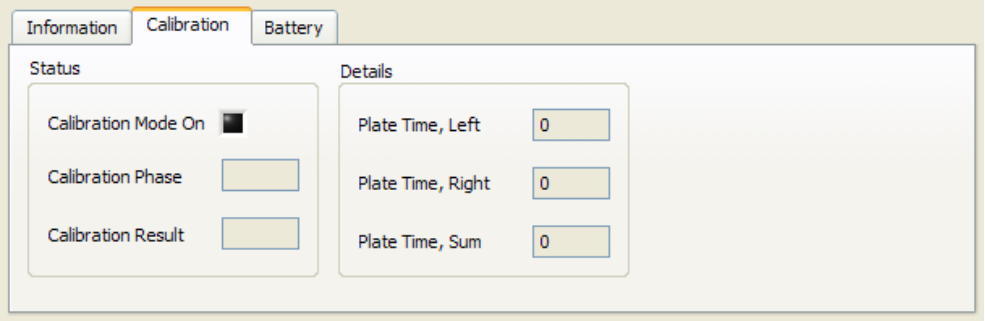
Keräsin WMC-ikkunan ensimmäiseen välilehteen pumppuun, vesisäiliöön, paineantureihin sekä sylkymaljaan liittyvät asiat, kuten kuvassa 16 näkyy. Muista moot-

Kuva 13: HERCO-kortin virranmittausnäkymä



Kuva 14: REFCO-kortin yleisnäkymä

toreista poiketen sylkymaljan kalibrointi tapahtuu automaattisesti, joten kalibrointimerkkivalolle ei sen tapauksessa vaikuttanut olevan välttämätöntä tarvetta. Vesisäiliön täyttöasteesta kertova tieto esiintyy kahdessa eri viestissä, joista ohjelma käsittelee molemmat. Tällä tavoin tieto päivittyy useammin, mikä näkyy käyttäjälle



Kuva 15: REFCO-kortin kalibrointinäkymä

parempana vasteena.

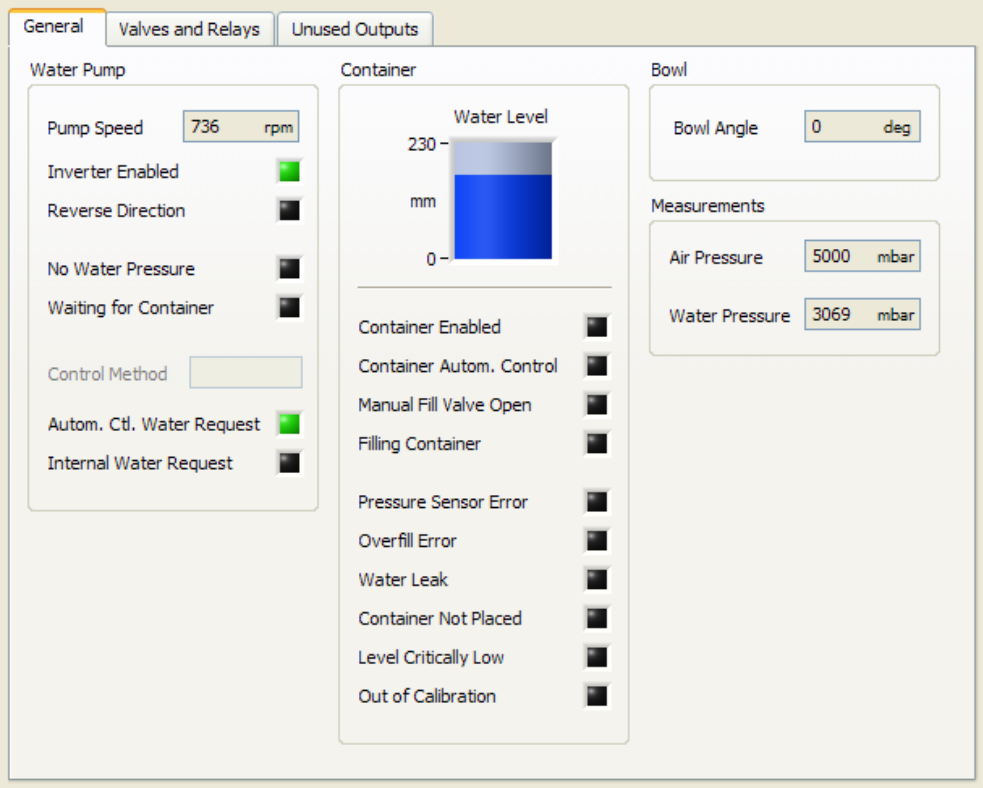
Paineanturien tiedot esitetään tällä hetkellä sellaisinaan, mutta jatkossa voisi tulla kyseeseen huomauttaa jollain tavalla huolestuttavan korkeista arvoista. Pelkän numeerisen tiedon perusteella käyttäjä ei välttämättä yhtä nopeasti havaitse mahdollista ongelmatilannetta. Painetietojen ilmaisemiseen voitaisiin myös käyttää jotakin toisentyyppistä graafista indikaattoria, josta lukeman merkitys käy paremmin ilmi. Nämä ovat kuitenkin asioita, joita on parempi hienosäätää vasta käyttökokemusten ja palautteen perusteella.

Venttiilien ja releiden tilat on koottu kahteen seuraavaan välilehteen. WMC-kortti voi ohjata kolmattakymmentä venttiiliä, mutta tyypillisesti näistä on käytössä vain osa. Yritin poimia ensimmäiseen venttiilitiloihin liittyvään välilehteen, joka näkyy kuvassa 17, oleellisimmat ja jätin loput kolmanteen välilehteen. Tämä vähentää hieman ruuhkaa näytöllä mutta mahdollistaa silti tarvittaessa kaikkien venttiilien seurannan.

Ultraäänianturien ja joidenkin muiden ominaisuuksien käsittely näkymästä jää vielä tässä vaiheessa puuttumaan, mutta täydennystä voidaan tehdä kun tarvetta ilmenee.

4.2.8 ICON

Instrumenttien ohjaus on monimutkaisimpia yksittäisiä ominaisuuksia hoitoyksikössä, sillä instrumentteja on valtavasti erilaisia ja kaikilla on omat erikoisuutensa. Ihanteellisessa tapauksessa kaikista instrumenteista saataisiin kerättyä kaikki mahdollinen tieto, kuten kulloinkin käytettävän instrumentin pyörimisnopeus/teho, vedenkäyttö, ilmankäyttö ja niin edelleen. Tämä saattaa kuitenkin käytännössä olla liian hankalaa toteuttaa, sillä suuri osa näistä tiedoista on pääkortin ja kosketusnäytön sisäisiä asioita. Kaikkien viestien lukeminen ja tulkitseminen edellyttäisi että diagnostiikkaohjelmaan toteutetaan merkittäviä osia sekä pääkortin että kosketusnäytön instrumenteista vastaavista toiminnoista, mikä olisi erittäin työlästä. Lisäksi molempien korttien ohjelmistojen toimintaa päivitetään ja muutetaan jatkuvasti, mikä tekisi diagnostiikkaohjelmiston ylläpidosta hyvin vaikeaa.



Kuva 16: WMC-kortin yleisnäkymä

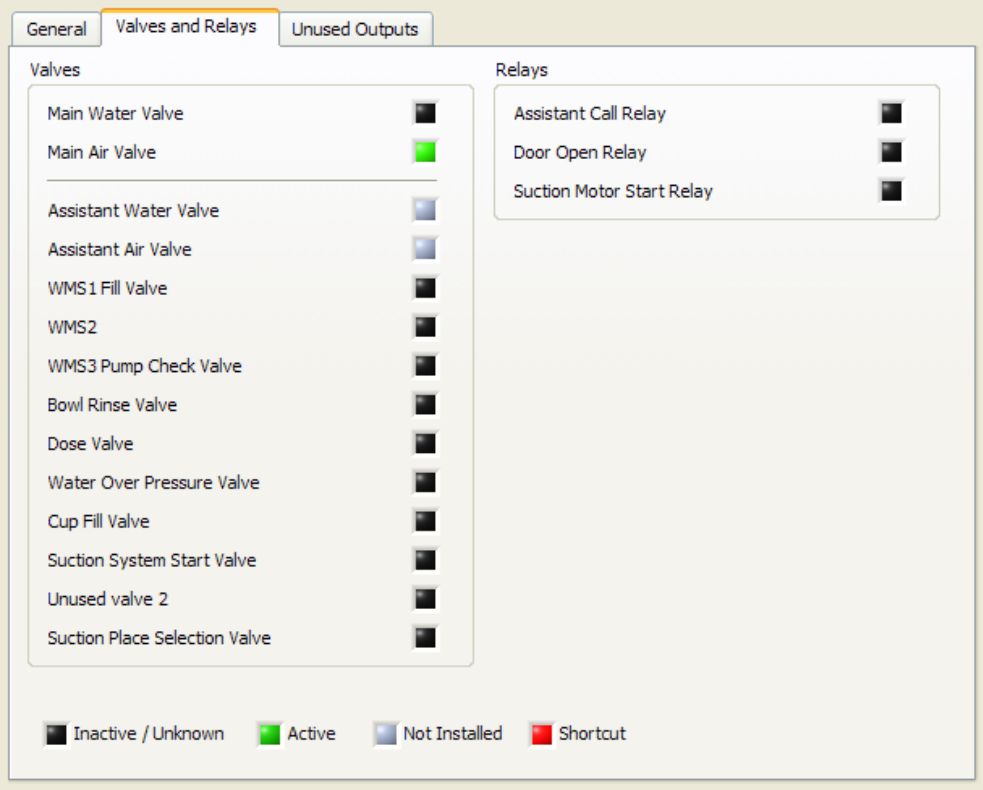
Toistaiseksi ICON-kortin osalta onkin siis toteutettu vain instrumenttien käytön seuranta yleisellä tasolla. Kuvassa 18 näkyy aikajanalla kunkin instrumenttipaikan tila viimeisen kymmenen minuutin ajalta. Tarkemmat tiedot instrumenttien käytöstä on järkevämpää toteuttaa joskus tulevaisuudessa oman, tarkoitukseen suunnitellun rajapinnan välityksellä.

ICON-kortin ohjaamat venttiilit voisi tuoda esille samaan tapaan kuin WMC:n näkymässä. Ensin täytyy kuitenkin miettiä, onko venttiilit parempi lajitella korttikohtaisiin näkymiin vai kerätä kaikkien venttiilien tilat johonkin yksittäiseen näkymään. Tämä on päätös jota on syytä käsitellä ajan kanssa, käyttäjiltä saadun palautteen perusteella.

4.2.9 Loput toimilaitteet

Muutamalle toimilaitteelle ei vielä tässä vaiheessa luotu minkäänlaista näkymää. Nämä laitteet ovat RETU, GUI, SUCO sekä valaisimet SLED ja OLCO.

RETU on langattoman jalkaohjaimen hoitoyksikönpuoleinen lähetin/vastaanotinkortti. Se välittää jalkaohjaimelta radioteitse saapuvat viestit CAN-väylälle eikä tee oikeastaan mitään muuta. Diagnostiikan kannalta kortti olisi silti kiinnostava kohde, sillä



Kuva 17: WMC-kortin venttiili- ja relenäkymä

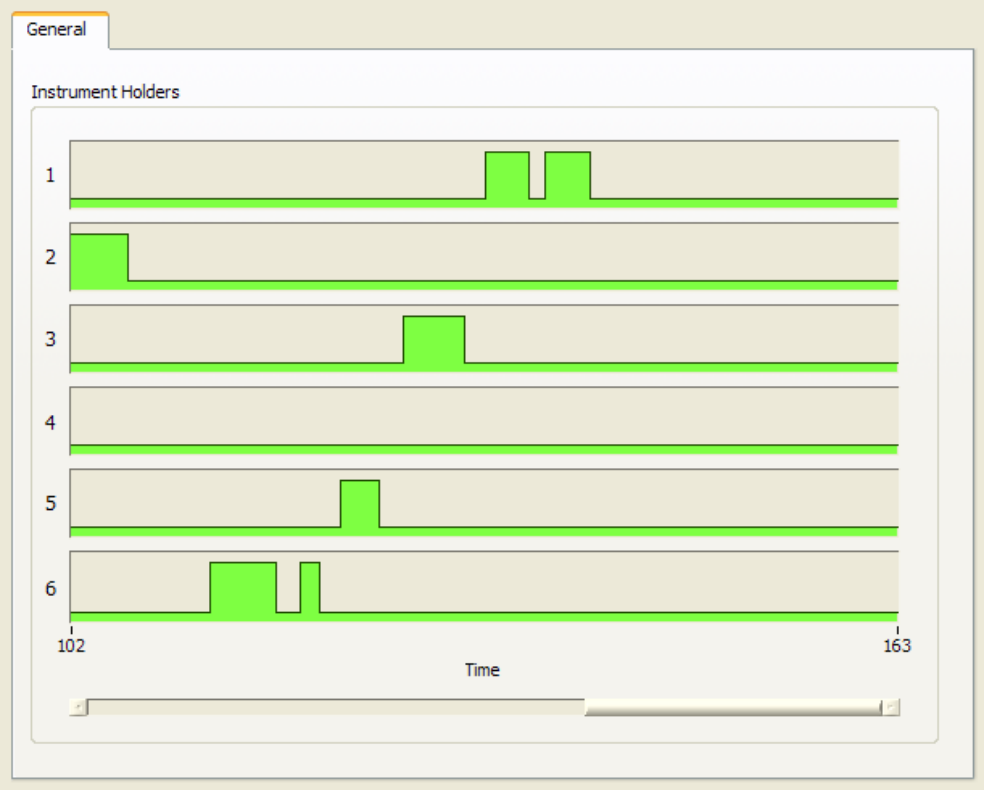
langaton yhteys tuo mukanaan omat ongelmansa, joita on välillä hankala selvittää. Toisinaan esimerkiksi kaksi samalla radiotaajuudella toimivaa langatonta jalkaohjainta häiritsee toistensa toimintaa, mutta tällä hetkellä hoitokoneen kautta ei ole mitenkään yksinkertaiesti nähtävissä, millä taajuuksilla on kuinkakin paljon liikennettä.

RETU-kortti kykenisi hyvin skannaamaan eri taajuuksia ja etsimään joukosta sellaisen, jolla on vähiten liikennettä. Suurin este radioyhteyteen liittyvien asioiden diagnostiikan toteuttamiselle on jälleen diagnostiikkaohjelmiston ja hoitoyksikön välisen viestiliikenteen yksisuuntaisuus. Koska RETU toimii lähes täysin passiivisesti ja välittää vain jalkaohjaimen viestejä, sen toiminnassa ei ole paljoakaan seurattavaa, ellei korttia voida ensin ohjeistaa vaihtamaan kanavaa tai tekemään jotakin muuta. Parannukset näihin asioihin jäivät kuitenkin tämän diplomityön ulkopuolelle.

GUI:n eli kosketusnäytön osalta diagnostiikalle ei ole esitetty mitään toiveita. Tämä voi johtua siitä että sen toimintaa on joka tapauksessa helppo seurata suoraan näytöltä. Tarvittaessa jonkinlainen diagnostiikkanäkymäkin voidaan toki tehdä.

SUCO:n osalta voisi tulla kyseeseen jokin vastaavankaltainen näkymä kuin ICON:illa on; erityisiä toiveita ei SUCO:n suhteen tullut suunnitteluvaiheessa esille.

Valaisimet ovat toiminnaltaan melko yksinkertaisia laitteita, eikä niiden toiminnassa



Kuva 18: ICON-kortin yleisnäkymä

tiettävästi ole esiintynyt juuri vikoja. Omat näkymät niillekin voidaan silti tarvittaessa tehdä, jotta on mahdollista seurata esimerkiksi valon kirkkautta sekä joitain muita perusasioita.

4.3 Käyttö tuotekehityksessä

Tuotekehityksen tilanne on diagnostiikkamahdollisuuksien osalta ollut kohtuullinen, mutta tälle uudellekin ohjelmistolle löytyy jonkin verran käyttöä. Käyttökokemuksia ei ehtinyt kertyä tässä kerrottavaksi vielä niin paljon kuin olisin toivonut, mutta seuraavassa kuitenkin joitain huomioita.

4.3.1 Tuotekehitys

CAN-viestien näkemisestä on hyötyä päivittäisessä kehitystyössä, etenkin sellaisten toimilaitteiden tapauksessa jotka eivät voi viestiä ulkomaailman kanssa sarjaportin tai muun vastaavan kautta. LabVIEW'lla on myös nopea rakentaa esimerkiksi uusi väliaikainen näkymä jonkin ominaisuuden testaamiseen, kun tarpeelliset kehysrakenteet ovat jo olemassa.

Selkeä käyttöliittymä mahdollistaa paremmin ohjelman käytön myös muille kuin ohjelmistosuunnittelijoille. Jonkin verran kiinnostusta ohjelman käyttöön onkin jo tullut muun muassa mekaniikkasuunnittelijoiden taholta.

4.3.2 Testaus

Ohjelmistotestauksessa diagnostiikkatyökalulla voidaan esimerkiksi tarkkailla että hoitoyksikön ohjelmiston toiminnassa ei näy muutoksia, joita ei kuuluisi olla. Eri näkymistä tallennettavilla kuvilla voidaan helposti verrata esimerkiksi jonkin moottorin virrankulutusta kahden eri ohjelmistoversion välillä.

Diagnostiikkaohjelmistoa on myös käytetty mekaanisten osien kestotesteissä. Jättämällä esimerkiksi niskatuki ajamaan jatkuvasti jotakin testisekvenssiä, voidaan seurata kasvaako moottoreiden virrantarve ajan myötä tai alkavatko asentoanturit pätkiä.

4.4 Käyttö tuotannossa

Tuotannossa diagnostiikkaohjelmistoa voidaan käyttää apuna sekä osakokoonpanojen että lopullisen hoitoyksikön testaamisessa. Valitettavasti ohjelmisto ei tätä kirjoitettaessa ollut aikomuksista huolimatta ehtinyt kovin laajaan käyttöön, joten varsinaisista käyttökokemuksista ei ole vielä paljoa kerrottavaa.

4.4.1 Osakokoonpanojen testaus

Etenkin potilastuolin kohdalla moottorien ja potentiometrien testaus ennen tuolin asentamista muuhun hoitoyksikköön on hyödyllistä, jotta mahdollisen ongelman ilmetessä ei tarvitse purkaa aivan koko laitetta osiin. Käytännössä tarkoitus on tarkistaa moottorien samanlaisissa olosuhteissa kuluttamia virtoja ja yrittää seurannan perusteella havaita ajoissa, jos jonkin yksilön virrankulutus on poikkeavan korkeaa. Korkea virrankulutus viittaa yleensä rikkonaiseen moottoriin tai liian tiukkaan mekaaniseen rakenteeseen.

Potentiometrien osalta osakokoonpanoa voidaan ajaa testisekvenssillä jonkin aikaa ja tarkastella, pätkiikö potentiometrin lähettämä signaali. Mahdollisen ongelman havaitseminen tuotannossa säästää paljon hankalammilta korjaustoimenpiteiltä siinä vaiheessa, kun hoitoyksikkö on jo asiakkaalla asti ja ongelma ilmenee käytössä.

4.4.2 Lopputestaus

Kun koko hoitoyksikkö on koottu lopulliseen muotoonsa, suoritetaan vielä kattava testaus. Testistä voidaan tallentaa esimerkiksi sopiva loki ja tutkia tuloksista poikkeaako jokin seikka merkittävästi keskimääräisistä tuloksista. Käytännössä tällaista käyttöä ei ole vielä kokeiltu.

4.5 Mahdollinen käyttö huoltotöissä kentällä

Diagnostiikkaohjelmistoa ei vielä tätä kirjoitettaessa ole ehditty laajamittaisemmin hyödyntämään huoltotöissä. Lyhyen kokeilun ajan eräältä klinikalta kerättiin käyttölokia hoitoyksikön viestiliikenteestä, mikä onnistui hyvin. Yhteys hoitoyksikköön säilyi vaikka hoitoyksikkö välillä sammutettiin ja lokin perusteella oli mahdollista päätellä haluttuja asioita yksikön käytöstä.

5 Yhteenveto ja johtopäätökset

Tässä viimeisessä osassa tarkastellaan miten diplomityön tuloksena syntynyt ohjelmisto vastaa alkuperäistä suunnitelmaa ja miten sitä voisi jatkossa kehittää.

5.1 Lopputuloksen arviointi

Työn tavoitteena oli kehittää Sovereign-hammashoitoyksikön parissa työskenteleville väline, jolla laitteessa esiintyviä ongelmia on nopea paikantaa. Kokonaisuutena arvioiden diagnostiikkaohjelmisto saavuttaa mielestäni tavoitteen melko hyvin. Ohjelmisto on helposti lähestyttävä ja toteuttaa suurimman osan selvitystyössä esille tulleista toiveista.

Koska työ täytyi resursoinnin vuoksi tehdä erillisenä projektina, eikä osana hammashoitoyksikön ohjelmiston tuotekehitystä, joitain toimintoja ei ollut mahdollista tai järkevää toteuttaa. Ohjelmisto ei myöskään valmistunut niin nopeasti että se olisi ehtinyt toivotunlaajuiseen koekäyttöön, minkä vuoksi lopullista arviota ohjelmiston käyttökelpoisuudesta ei voida tässä työssä arvioida. Myös suunnitellut käyttöliittymän parannuskierrokset jäivät samasta syystä työstä pois. Pääosin palaute, jota ehti kertyä, oli kuitenkin positiivista.

Työhön valittu LabVIEW osoittautui hyväksi kehitysympäristöksi. Muutoksia oli mahdollista tehdä ja kokeilla hyvin nopealla vasteella. Kun ohjelman perusrakenne oli saatu kuntoon, uusia ominaisuuksia toimilaitenäkymiin oli mahdollista toteuttaa miltei sitä mukaa kuin jotain pyydettiin.

5.2 Puutteet ja jatkokehitys

Ohjelmiston jatkokehitystä suunnitellessa on ennen kaikkea tärkeää kuunnella siitä saatavaa palautetta. Jos mitään perustavanlaatuisia puutteita ei ilmene, seuraava askel on laajentaa tukea yhä puuttuville toimilaitteille ja lisätä käsitteljöitä tarpeellisille viesteille.

Käytettävyyden osalta suurin puute johtuu viestiliikenteen yksisuuntaisuudesta. Koska ohjelmisto ei voi suoraan kysyä hoitoyksiköltä haluamaansa tietoa, vaan joutuu odottelemaan että kyseinen viesti esiintyy väylällä, saattavat toimilaitenäkymissä esitettävät tiedot toisinaan olla vanhentuneita. Ongelmaa esiintyy eritoten sellaisissa tapauksissa joissa hoitokone on jo ollut käynnissä ennen diagnostiikkayhteyden ottamista. Tilannetta voisi lievittää liittämällä tärkeimpiin tietoihin ajastimia, jotta käyttäjä ainakin helpommin huomaisi tiedon olevan mahdollisesti vanhentunutta.

Siistimpi ratkaisu ongelmaan olisi saada hoitokone käyttäytymään diagnostiikkatilassa hieman eri tavalla ja päivittämään tietoja useammin. Pieni periaatteellinen ongelma tosin on, että järjestelmän toiminta tällöin muuttuu ja tutkittava ongelma saattaa teoriassa kadota.

Alemmalla tasolla pahin ongelma on ohjelman riippuvuus CAN-viestien tunnisteis-

ta ja niiden sisällön rakenteesta. Jos viestit muuttuvat, tietojen käsittely ja esitys täytyisi päivittää vastaavasti. Ongelmaan ei ole oikeastaan muuta kestävää ratkaisua kuin että hoitoyksikön ohjelmistolla olisi jokin yhtenäinen lähde CAN-viestien kuvaukselle, jota sitten myös diagnostiikkaohjelma voisi käyttää. Tällöin viestit tulkittaisiin automaattisesti oikein, kunhan tiedetään hoitokoneen ohjelmiston versionumero.

Diagnostiikan avulla on tarkoitus voida selvittää ongelmien syitä. Hyvän diagnostiikan avulla ongelmat selviävät helposti ja nopeasti. Uskon että tämän diplomityön tuloksena on saatu luotua perusta ohjelmistolle, jolla päästään lähelle tätä tavoitetta.

Viitteet

[1] Online Etymology Dictionary *diagnosis* http://www.etymonline.com/index.php?term=diagnosis (viitattu 20.3. 2010)

[2] Saman M I Adham, Ken Brough, Bruce Ecroyd et al. *Linking diagnostic software to hardware self-test in telecom systems* IEEE Communications Magazine, ISSN: 0163-6804 DOI: 10.1109/35.769278

[3] N. Mukherjee, T.J. Chakraborty, R. Karri *Built in self test: a complete test solution for telecommunication systems* IEEE Communications Magazine, volume 37, issue 6, June 1999 ISSN: 0163-6804 DOI: 10.1109/35.769277

[4] *Euroopan parlamentin ja neuvoston direktiivi 98/69/EY moottoriajoneuvojen päästöjen aiheuttaman ilman pilaantumisen ehkäisemiseksi toteutettavista toimenpiteistä ja neuvoston direktiivin 70/220/ETY muuttamisesta*, 1998, http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31998L0069:FI:HTML (viitattu 13.3. 2010)

[5] Manske et al. *United States Patent 4,898,263: Elevator Self-Diagnostic Control System*

[6] Messaros et al. *United States Patent 5,594,663: Remote Diagnostic Tool*

[7] Olaf Pfeiffer, Andrew Ayre, Christian Keydel *Embedded Networking with CAN and CANopen* päivitetty 1. painos Yhdysvallat, Copperhill Technologies Corporation, 2008, ISBN 978-0-9765116-2-5

[8] ISO 11898:2003, Road vehicles – Controller area network (CAN), 2003, http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=33422 (viitattu 11.3. 2010)

[9] Pat Richards *AN228: A CAN Physical Layer Discussion* (Application Note) Yhdysvallat, Microchip Technology Inc., 2002, http://ww1.microchip.com/downloads/en/AppNotes/00228a.pdf (viitattu 13.3. 2010)

[10] Robert Bosch GmbH *CAN-Specification, Version 2.0*, 1991, http://www.semiconductors.bosch.de/pdf/can2spec.pdf (viitattu 8.3. 2010)

[11] Phil Jones *Visual Basic: A Complete Course* 1st ed. Iso-Britannia, Martins the Printers Ltd, 1998, ISBN 0-8264-5405-4

[12] Gary W. Johnson, Richard Jennings *LabVIEW Graphical Proramming* 4th ed. Yhdysvallat, The McGraw-Hill Companies, 2006, ISBN 0-07-145146-3

[13] National Instruments *What Is NI TestStand?* Yhdysvallat, 2010, http://zone.ni.com/devzone/cda/tut/p/id/6073 (viitattu 15.3. 2010)

[14] Soren Lauesen *User Interface Design: A Software Engineering Perspective* Iso-Britannia, Pearson Education Limited, 2005

[15] Reinhard Oppermann *User-interface design* http://fit.fraunhofer.de/~oppi/publications/UserInterfaceLearningSystems.pdf (viitattu 5.4. 2010) DOI: 10.1.1.72.5696

[16] Edward R. Tufte *The Visual Display of Quantitative Information* 2. painos Yhdysvallat, Graphics Press LLC, 2006, ISBN: 0-9613921-4-2